Desarrollo de software con metodologías ágiles y Test-Driven Development aplicadas en el sector bancario en la compañía Accenture

SANTIAGO ALCAZAR CHAMORRO

Trabajo de grado en Modalidad de Práctica Profesional en Ingeniería de Sistemas

Director
Ing. Erwin Meza Vega

Asesor
Ing. Homero Nuñez Valencia

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Programa Ingeniería de sistemas
Popayán, Agosto de 2021

Desarrollo de software con metodologías ágiles y Test-Driven Development aplicadas en el sector bancario en la compañía Accenture

SANTIAGO ALCAZAR CHAMORRO

Trabajo de grado presentado a la Facultad de Ingeniería Electrónica y Telecomunicaciones de la Universidad del Cauca para la obtención del Título de Ingeniero de Sistemas

Director
Ing. Erwin Meza Vega

Asesor
Ing. Homero Nuñez Valencia

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Programa Ingeniería de sistemas
Popayán, Agosto de 2021

Agradecimientos

Quiero agradecer a Dios por darme la vida, y la oportunidad de vivirla feliz. Gracias a Él soy quien soy, y estoy donde estoy. A pesar de algunas pocas dificultades, siempre me mantuvo por el camino correcto de su mano. Me brindo a mi familia, los seres más importantes de mi vida, que en todo momento me apoyaron y han estado incondicionalmente conmigo desde siempre. Doy gracias a Dios por mi madre y mi padre, que juntos me dieron toda la confianza y los medios para lograr mis metas, su apoyo siempre fue incondicional. Igualmente agradezco a mis hermanos, cada uno de ellos aportó en gran medida en mi formación diaria, tanto académica como emocional, dándome a conocer nuevas experiencias y conocimientos que me permiten día a día construir mi futuro.

Doy gracias a mi Universidad, que me brindo el mejor conocimiento a través de excelentes docentes que me acompañaron a lo largo de la carrera permitiendo llegar a esta última etapa, seguro de que cada clase y concejo serán de mucha importancia en mi vida. Gracias al Ingeniero Erwin Meza Vega, director de la presente práctica profesional, quien me acompañó en este proceso, aportando desde su experiencia a mi crecimiento profesional.

Agradezco a la empresa Accenture, por confiar en mí y en mis conocimientos, dándome la oportunidad de pertenecer a los importantes proyectos donde aprendí demasiado. En la empresa amplié mis conocimientos y crecí como persona, gracias a que reconocen el valor de un empleado y a un excelente grupo de trabajo dirigido por Homero Nuñez, quien estuvo siempre al pendiente y dispuesto para poder desarrollar con éxito la práctica.

Finalmente agradezco a cada una de las personas que me apoyaron e hicieron posible que este gran largo se hiciera realidad, a toda la familia Alcazar y Chamorro, a todos los amigos de la Universidad con los que trabajamos fuertemente para estar aquí, y a mi comunidad Yeshua.

Este gran logro va dedicado a Dios y mi familia, por siempre estar acompañarme y confiar en mi. A todos ¡Mil gracias!

Contenido

Lista de figuras

Lista de tablas

Lista de acrónimos

TDD: Test Driven Development
NPM: Node Package Manager
VPN: Virtual Private Network
QA: Quality Assurance
CDN: Content Delivery Network
IDE: Integrated Development Environment
SaaS: Software as a Service
SM: Scrum Master
APN: Application Management
API: Application Programming Interface
REST: Representational State Transfer
SOAP: Simple Object Access Protocol
IoT: Internet of Things
SSH: Secure Shell
HU: Historias de usuario
DAO: Data Access Object
GAPI: Google API

Capítulo 1

1. Introducción

1.1 Contexto

Accenture es una empresa internacional creada en Reino Unido con su sede principal en Irlanda, dedicada a las tecnologías de la información en diferentes sectores de la industria, bajo intereses como la consultoría y outsourcing [1]. Cuenta con más de 537.000 empleados a nivel mundial, brindando a sus clientes valiosos proyectos caracterizándose por su cultura organizacional y valores, base de su actuar y toma de decisiones, grupales e individuales [1]. También se encuentra presente en industrias como la automotriz, sectores bancarios, aeronáutica, recursos naturales, seguros, software, entre otras [1]. En la industria del software, toma una importante posición del flujo de internet y la seguridad de la información prestando atención al sector bancario.

Así es como ha logrado tener a su cargo el 25% del trafico mundial de internet, procesar 300 millones de pasajes aéreos al año y proteger el 70% del correo electrónico a nivel global [2]–[4]. Dado a su excepcional control de la información, proyectos y personal, ha conseguido de manera consecutiva por 17 años, el premio a “Compañía admirable de Fortune”, añadiendo que se encuentra entre las 20 compañías más sostenibles del mundo, según Barron’s [5]. Esta compañía cuenta con sedes de operación en 50 países, entre ellos Colombia, ubicando sus oficinas de tecnología en las ciudades de Bogotá y Medellín.

Accenture Colombia brinda la oportunidad a profesionales colombianos para hacer parte de su empresa, conformando equipos de marketing, finanzas, desarrollo de software, entre otros. En esta última área, brinda a estudiantes de últimos semestres, la posibilidad de integrarse como practicantes en el área profesional. Cada seis meses, abre convocatorias a estudiantes de carreras universitarias, aplicando filtros que no solo consideran las áreas técnicas de trabajo, sino también las habilidades blandas, técnicas de comunicación y trabajo en equipo.

De esta forma se obtuvo la posibilidad de pertenecer durante cinco meses a la empresa ejerciendo tres roles fundamentales, los cuales implicaban desarrollar software como labor principal, y apoyar en variadas actividades de gestión de proyecto. Se realizó el trabajo con dos de los bancos más importantes del país, el banco Icol y el banco Cubo, con los cuales se han trabajado varios proyectos relacionados a la tecnología de la información desde hace unos años, impulsados por un proceso de transformación digital propuesto por Accenture y cada cliente, para cambiar aspectos internos y externos de cada organización.

En el inicio de la práctica, el grupo que dirigía el trabajo con el cliente Icol realizó el respectivo posicionamiento de cada practicante analizando su perfil y experiencia, y posterior a algunas capacitaciones guiadas por la empresa, se desarrollaron las funcionalidades solicitadas, que correspondían a historias de usuario o tareas, las cuales se actualizaban frecuentemente en la herramienta de gestión Jira. Todo ello, acorde a las necesidades que tenía el proyecto en este momento para ser culminado satisfactoriamente. Acorde al rol de desarrollador de software, se observó que la necesidad principal se encontraba en la ejecución de cada funcionalidad, que en su mayoría correspondían a detalles para finalizar la aplicación móvil, más exactamente los componentes WebView. Todo ello, relacionado a la interacción del usuario, validaciones, errores expuestos por el equipo de QA y pruebas de la aplicación desde el rol establecido.

Posteriormente, con motivo de finalización del proyecto con el banco Icol, se efectuó un traslado para realizar actividades relacionadas a la gestión de proyectos con el banco Cubo en donde se requería apoyo, tanto para realizar monitoreo de cronogramas, levantamiento de actas, entre otros, como para innovar a través de herramientas proporcionadas por la Suite de Google y agilizar ciertos procesos relacionados con la presentación de evidencias de trabajo.

Finalmente se desarrolló una aplicación (Accenture Forms) como solución a un problema identificado que será desglosado en apartados posteriores, para poder gestionar la información que se mostraría en cada avance de una mejor forma, guardado esta información para ser consultada por Google Data Studio que sería la herramienta por la cual se monitorea cada proyecto o incidencia, con su respectivo progreso.

Es importante resaltar que las metodologías agiles y el desarrollo guiado por pruebas (TDD) representan algunas de las herramientas que se usan en la actualidad para llevar a cabo proyectos de desarrollo de software, asíٔ es como, las empresas han encontrado en ellas un recurso valioso para aprovechar de forma eficiente el talento humano y los recursos disponibles, ayudándoles a lograr sus objetivos y cumplir con las necesidades de los clientes [6], [7]

Durante este tiempo y acorde a las indicaciones de la empresa, se aplicó la metodología y la práctica de desarrollo mencionadas anteriormente, con el fin de organizar el trabajo a realizar y aportar calidad a los productos, intentado que los clientes se encuentren satisfechos con cada Sprint realizado, que hace parte de la entrega final. Esto se aplica para el desarrollo de la aplicación móvil, donde había un Scrum Master que verificaba que los procesos se llevaran a cabo correctamente, y en el acompañamiento y supervisión del programador Senior para realizar el desarrollo guiado por pruebas.

De la misma manera, fueron pilares fundamentales para llevar a cabo el desarrollo del aplicativo Accenture Form, donde en relación a las metodologías ágiles, en el marco de trabajo Scrum, se realizaron 7 sprint, trabajando un *backlog* en conjunto con el cliente y sus preferencias o necesidades, y realizando monitoreos y entregas constantes, todo ello en el marco del alcance expuesto en un documento inicial. En cuanto a la codificación, se realizó el desarrollo guiado por pruebas acorde a lo aprendido con el ingeniero del banco Icol, lo cual fue de gran ayuda para entregar un código funcional y refactorizado.

Finalmente fue posible apoyar a las necesidades de cada uno de los clientes con los que trabaja la compañía. Para el banco Icol, desarrollando y probando cada funcionalidad que hacía parte de la aplicación móvil para entregar el proyecto, y respecto al banco Cubo gestionando herramientas y creando soluciones innovadoras para agilizar procesos de gestión de proyectos.

1.2 Objetivos

En los siguientes apartados se presenta el objetivo general y los objetivos específicos aprobados en el marco de la presente práctica profesional.

1.2.1 Objetivo general

- Apoyar el equipo de transformación digital de la empresa Accenture en la construcción de funcionalidades destinadas a aplicaciones del sector bancario usando metodologías agiles y TDD.

1.2.2 Objetivos específicos

- Caracterizar el estado del proyecto de transformación digital de la empresa Accenture respecto a las metodologías usadas, la conformación del equipo de trabajo y el avance actual del proyecto.
- Participar y contribuir en las actividades de gestión del proyecto de acuerdo con las practicas establecidas por Accenture.
- Implementar, probar y documentar las funcionalidades asignadas por la coordinación del equipo de trabajo siguiendo las practicas metodológicas definidas en el contexto de la empresa y los conocimientos y habilidades desarrolladas en la carrera.
- Recopilar y analizar la experiencia desarrollada en la practica desde la perspectiva de desarrollo de nuevas habilidades e incorporación en un entorno empresarial.

1.3 Marco conceptual

En la siguiente sección se evidencian los conceptos a tener en cuenta para el desarrollo de la presente práctica profesional.

1.3.1 Metodologías Ágiles

Las metodologías ágiles son aplicadas en la actualidad en una gran cantidad de proyectos de desarrollo de software. Estas metodologías se apoyan en el usuario final, permitiendo que se involucre en todas las etapas del desarrollo, dando la posibilidad que algunos requerimientos cambien [8]. Se realizan pruebas a lo largo de todo el proyecto y se integran funcionalidades de forma iterativa, creando prototipos que son presentados al cliente en cada iteración, produciendo así requisitos acordes a las necesidades del cliente, en pro de la constante comunicación [8].

Es posible afirmar que las metodologías ágiles son aplicadas cuando se responde rápidamente al cambio, con el fin de aumentar la calidad del producto y la satisfacción del cliente. Además, cuando se cumplen con estas cinco fases: Definir y analizar los requisitos, diseñar, implementar, probar, presentar o lanzar el conjunto de funcionalidades desarrolladas y por último, el mantenimiento [9].

Las metodologías convencionales como cascada hacen que el usuario interactúe en el principio del proyecto, tratando de obtener todos los requerimientos, pero a medida que el proyecto avanza, la comunicación se va perdiendo a tal punto que el cliente no conoce en qué estado se encuentra su producto [8], por esto, las metodologías ágiles tratan de solucionar este problema incluyendo lo más prudente y adecuado, al cliente en el equipo.

Existen algunas ventajas y características sobre las metodologías ágiles que menciona [8], algunas de ellas son:

- El cliente tiene una participación activa a lo largo del proyecto, por ende, tendrá una amplia idea del producto final.
- Los clientes no necesitan esperar al final para obtener su producto, ya que el desarrollo por ciclos permite entregas constantes.
- Existe una alta tasa de satisfacción de los clientes al utilizar esta metodología.

La aplicación de las metodologías ágiles puede darse en equipos de cualquier tamaño, sin embargo, puede ser un reto para equipos de muchos integrantes, ya que la comunicación interna influye respecto al desarrollo cíclico [10]. La priorización y asignación de tareas y mantenimiento de la calidad, son factores que se deben considerar. Además, circunstancias como la exagerada participación del cliente en el proyecto también debe realizarse con el mayor profesionalismo, para que este, no se convierta en un causante de retrasos [10].

Scrum, Extreme Programming, Crystal, RAD, entre otras, son marcos de trabajo que hacen parte de las metodologías ágiles. Como lo indica [9], Scrum es considerada la mejor opción para construir aplicaciones de calidad, y cada una de ellas, tiene alguna especialidad, como por ejemplo, para RAD son los proyectos menos complejos con requisitos claros.

1.3.2 Test-Driven Development (TDD)

Las constantes investigaciones sobre la ingeniería de software ponen a prueba las diferentes propuestas que realizan los investigadores, por ello, las nuevas metodologías, técnicas y prácticas, con sus respectivos casos de estudios son frecuentemente cuestionados, para mostrar sus beneficios y aumentar la confiabilidad de los hallazgos [11]. TDD es una práctica para desarrollar software, en la cual se implementan pequeñas funcionalidades en cada ciclo de desarrollo [12], los cuales están conformados por:

1. El desarrollador crea, antes de iniciar a programar el requerimiento, los casos de prueba para cerciorarse que funcione correctamente.
2. Se ejecutan las pruebas, dando así el primer fallo. Puede servir para evidenciar que los casos están correctamente escritos.
3. Desarrollar la mínima cantidad de líneas de código, con el fin de aprobar todas las pruebas planteadas en el punto 1. En caso de alguna fallida, se modifica el código hasta que sean exitosas.
4. Refactorizar el código para aportar calidad.
5. Regresar al paso 1.

Los anteriores pasos los describen [13], donde se realiza un estudio experimental para unir esta práctica de desarrollo con la mutación, la cual está orientada al testing. De esta manera se puede afirmar que TDD no es una práctica para desarrollar pruebas funcionales. Aporta casos de prueba, pero estos son utilizados únicamente por los desarrolladores, ya que al ser enviados a los *testers*, podrían generar un sesgo afectando ciertas perspectivas y habilidades. TDD es el resultado de la unión entre first-test development (FTD) y la refactorización de forma continua y cíclica [7].

En [14], se realiza una investigación experimental, la cual parte de las siguientes preguntas: ¿Otorga TDD calidad al producto final? ¿Están los desarrolladores satisfechos con la aplicación de la práctica? Luego de tomar varios estudios empíricos, extraer estadísticas de los desarrolladores y demás procesos, se concluye que la práctica es bastante novedosa, sin embargo la eficacia es poco concluyente, ya que estudios muestran la gran utilidad de TDD, sin embargo, algunos otros, muestran que el impacto es nulo o menor, que el desarrollo convencional.

En metodologías ágiles, la inclusión de Test-Driven Development es recomendada como factor principal para reducir costos y mejorar la calidad del código escrito [12]. Como lo menciona [15], las metodologías ágiles, se desvían de los enfoques tradicionales que han venido impactando la ingeniería de software y sus nuevas perspectivas, de esta forma, realizan una comparación entre el desarrollo basado en pruebas y el desarrollo basado en comportamientos (TDD y BDD respectivamente). Concluyendo así, en cuanto a *software* que se encuentra en condiciones reales de producción, que aplicar TDD aporta más calidad al producto entregado y mayor calidad interna ante el grupo de trabajo.

Dado que esta práctica puede guiar y ayudar a los desarrolladores en las pruebas unitarias, podríamos decir que el desarrollo se efectúa más rápidamente, siempre y cuando los desarrolladores cuenten con experiencia para poder plantear estas pruebas antes del desarrollo, sumado a ello, la generación del *feedback* inmediato, específico y local, puede tener efectos positivos sobre la calidad del código y la velocidad en la que se desarrolla [12], hechos que son relevantes para las metodologías ágiles.

1.3.4 Transformación digital

Actualmente, nos encontramos en una cuarta revolución industrial, provocada por el internet, dando como resultado diversas investigaciones para aprovechar al máximo su uso, de esta forma, es como las empresas se han interesado más por automatizar sus procesos, y tratar de mejorar sistemas que a través de los años, se han convertido en obsoletos [16]. Computación en la nube, algoritmos inteligentes para transformar negocios y procesos basados en computadores son algunos ejemplos de requerimientos que se evidencian, para que las empresas se mantengan competitivas en el mercado [17].

De esta manera, se podría decir que la transformación digital está ocurriendo en diferentes industrias, cuya base es la ingeniería de software, con áreas como el internet de las cosas (IoT), que favorece la incorporación de estas tecnologías en nuestro día a día [18]. Podríamos definirlo entonces como el proceso mediante el cual, las empresas buscan a través de soluciones tecnológicas, hacer más eficientes sus procesos, esto con el fin de proporcionar en sus productos o servicios, resultados de valor a los clientes [19], involucrando soluciones acordes a las demandas del mercado actual.

Como lo indica [19], es importante que una transformación digital refuerce y amplíe la visión de una organización, tratando de disminuir los costos de los procesos, migrando sus tecnologías a servidores en la nube apropiados para cada proyecto, lo que permitirá una mejor gestión de la información para capturar datos y otorgar una rápida y acertada respuesta a los clientes.

1.3.4 Aplicaciones Web y Móviles

A medida que la tecnología avanza, el sector bancario igual que otras industrias ha renovado tanto sus procesos internos como externos, con el fin de brindar un mejor servicio y automatizar procesos para obtener mayores beneficios, a través de soluciones tecnológicas (aplicaciones web, aplicaciones móviles, entre otros) que se basan en el internet [20], [21]. Colombia es uno de los principales países exportadores de aplicaciones móviles en América Latina y a pesar de la inestabilidad económica se espera que este mercado continúe creciendo, fabricando cualquier tipo de aplicaciones para clientes internacionales, brindando empleo en el país [22].

Las aplicaciones móviles resultan una alternativa para ofrecer a los clientes diversos servicios de la compañía sin necesidad de dirigirse presencialmente a un establecimiento [20]. Estas pueden ser:

- Aplicaciones nativas: Las aplicaciones nativas consumen un API proveído por el sistema operativo del dispositivo, por medio del cual es posible acceder a un conjunto de recursos mediante un modelo de permisos definido [23].

- Aplicaciones embebidas: Son el resultado de una mezcla entre una aplicación web y una aplicación nativa, ya que a través de un componente WebView insertan en determinadas ubicaciones un navegador en la aplicación sin la necesidad de desarrollar toda la aplicación como una página web [23], [24].

Cabe aclarar que existen otro tipo de aplicaciones móviles, tales como web móviles o híbridas, las cuales pueden igualmente adaptarse a ciertas necesidades.

Las aplicaciones web también representan una solución frente a las necesidades actuales del mercado, las cuales deben evaluarse como productos software complejos [25]. A través de una aplicación web es posible acceder a una enorme cantidad de servicios y recursos, los cuales pueden ser multimedia, de almacenamiento o incluso, actividad bancaria. Por esta razón, para responder ante la diversidad de servicios, los navegadores han evolucionado para convertirse en un entorno de ejecución de aplicaciones web, siendo capaces de ofrecer interacciones con el usuario a través de cuadros de diálogo, notificaciones, ventanas emergentes, carga de scripts, entre muchos otros, para que servidores ubicados en cualquier parte del mundo procesen la información [26].

Para desarrollar una aplicación web existen diversas tecnologías que se utilizan en la actualidad, una de ellas es React, la cual es una biblioteca de Javascript creada por Facebook para desarrollar interfaces de usuario multipropósito tan complejas como el usuario lo requiera. React no se centra únicamente en aspectos web, ya que, en conjunto con otras librerías, puede usarse para crear aplicaciones móviles o aplicaciones de realidad virtual [27].

Para construir una aplicación web es necesario considerar diversos temas en cuanto al desarrollo, uno de ellos es la gestión de estados, que es una característica primordial para construir aplicaciones móviles o web escalables [28]. Redux es una biblioteca de administración de estado predecible para administrar el estado global de una aplicación Javascript, la cual es ampliamente utilizada en aplicaciones construidas con React; Se basa en el patrón de diseño Flux construido por Facebook igualmente, en donde interactúan cuatro componentes principales: *store, dispatcher, actions* y *view* [28].

Uniendo las bibliotecas mencionadas anteriormente, es posible desarrollar aplicaciones que se ajusten a las necesidades del mercado actual. Estas aplicaciones pueden ser gestionadas por un SaaS (Software as a Service), el cual brinda diferentes servicios en la nube tales como administrar hardware, middleware, despliegues y seguridad informática, permitiendo a los usuarios reducir costos, escalar y actualizar aplicaciones más rápidamente, además de estimar el costo final del proyecto con mayor precisión [29]. Firebase y Vercel son algunos ejemplos de SaaS que existen en la actualidad. Firebase por su parte pertenece a Google y brinda servicios como hosting, pruebas automatizadas, base de datos, autorizaciones, entre otros.

Capítulo 2

2. Caracterización de los proyectos

La caracterización de los proyectos se llevó a cabo desde diferentes ámbitos. Inicialmente era necesario apoyar el proyecto del banco Icol, en el cual participaron alrededor de 30 personas: Administrativos, con un enfoque de tecnología, Ingenieros en áreas como ingeniería administrativa, ingeniería industrial, e ingeniería de sistemas, líderes técnicos y líderes de proyecto que tomaban decisiones sobre el producto.

El rol desempeñado en este proyecto se orientaba hacia el desarrollo, por ello la caracterización se llevó a cabo partiendo de los permisos otorgados y la interacción o comunicación con el grupo de trabajo. Debido a que los cargos y las responsabilidades estaban determinados de forma estricta, desde el rol desempeñado no se conocía en detalle la forma como se estaban desarrollando otros subproyectos.

Posteriormente se desarrollaron actividades en el marco del Banco Cubo. En este contexto se realizó la caracterización de algunos procesos de gestión del proyecto, específicamente, la entrega semanal de evidencias o actividades en proceso, que se mostraban a los delegados del banco. En este caso no existía una metodología o un proceso estructurado para mostrar al cliente los avances realizados, por ello, se buscaba implementar mecanismos para mejorar la presentación de los resultados.

2.1 Caracterización banco Icol

El banco Icol es un cliente de Accenture desde hace algunos años, han ejecutado varios proyectos de la mano de la compañía y para el momento en el que se inició la práctica profesional, se encontraba en ejecución el desarrollo de un conjunto de aplicativos, para lo cual se tenía autorización participando como programador.

2.1.1 Estado del proyecto

El proyecto sobre el cual se apoyó en el desarrollo consiste en la realización de dos aplicaciones con las mismas características y funcionalidades para los dos sistemas operativos más importantes en la actualidad: iOS y Android. Presentaba un leve retraso según lo expresado por las personas encargadas de la gestión del proyecto, por condiciones de fuerza mayor, sin embargo, el proyecto se encontraba en su etapa final. Estas dos aplicaciones unían sus funcionalidades cuando se realizaba el llamado a las WebView para realizar una transacción.

En el módulo de “retiros sin tarjeta”, subproyecto en el cual se desarrollaron todos los requerimientos a cargo, ya se encontraban realizadas todas las pantallas WebView (páginas web) que involucraba esta sección del proyecto, por ende, el desarrollo fue orientado a ajustes en la lógica, cambios en la interfaz, pruebas unitarias y de integración, y solución de errores expuestos por el equipo de QA. Cada subproyecto estaba conformado por un grupo de desarrolladores, y en el proyecto trabajaban profesionales de diferentes disciplinas, que se encargaban desde la gestión del proyecto hasta aspectos financieros y económicos del mismo, que eran tratados con el cliente.

2.1.2 Autorizaciones

Dado que el nivel de seguridad que implica trabajar con un cliente del sector bancario es alto, para poder participar en el proyecto fue necesario esperar la asignación y entrega de un equipo de cómputo y las credenciales necesarias para acceder a la red de la empresa mediante una Red Privada Virtual (VPN) y un token de seguridad enlazado a un teléfono y al correo de la compañía.

Luego de obtener los recursos para acceder a los sistemas del cliente, era necesario reportar al líder técnico de acuerdo con el subproyecto al cual pertenecería y dependiendo si se iba a desarrollar para front-end o back-end, la gestión para obtener el acceso a los repositorios. Inicialmente se requería solicitar al cliente un usuario temporal en el sitio web donde se alojaban los repositorios. De acuerdo con los permisos asignados, se podía acceder únicamente al código fuente de la rama o subproyecto al cual se pertenecía.

Es importante anotar que la autorización tomó un tiempo considerable por la eventualidad de la pandemia, ya que en su mayoría estas peticiones se realizaban presenciales. Para solucionar esta situación, se implementaron mecanismos de comunicación vía correo electrónico. Todas las actividades a realizar referentes a los permisos quedaban registradas en actas para el cliente, indicando que el empleado de Accenture iba a acceder a determinado repositorio o repositorios por un lapso de tiempo determinado, únicamente con el usuario asignado.

Finalmente se configuró el acceso por Shell Seguro (SSH) y el Gestor de Paquetes de Nodejs (NPM), ya que el acceso a internet por medio del equipo accedido remotamente era limitado y se debía configurar para obtener las respectivas librerías utilizadas.

Al participar directamente en el proyecto se pudo comprobar que el nivel de seguridad que brinda Accenture a sus clientes es alto, respecto al almacenamiento, el acceso y la gestión de la información, ofreciendo total confianza y siguiendo restricciones de seguridad desde las más sencillas a las más complejas. Es importante anotar que no

solo se aplica la seguridad en el software o producto que se está construyendo, sino a todos los procesos de gestión que involucran al cliente y su información.

Dadas las restricciones de seguridad establecidas para la gestión de los proyectos, no se tuvo información completa referente a la organización de los equipos de trabajo que permitiera constatar que todos los proyectos se llevan a cabo siguiendo los mismos parámetros. No obstante, fue posible observar que el proyecto de la aplicación se dividía en subproyectos, cada uno de los cuales era atendido por equipos de trabajo de alrededor de 8 personas.

2.1.3 Estructuración del equipo

Cada subproyecto tenía a cargo un módulo o funcionalidad de la aplicación, que podía pertenecer, por ejemplo, a un flujo de una transacción. Para el subproyecto al cual se perteneció, el grupo de trabajo constaba de 8 personas, 4 de ellas desarrollando en tecnologías back-end y 4 en tecnologías front-end. En el proyecto existía un participante que conocía el estado de varios subproyectos, y se encargaba de interactuar con el cliente y mostrar a través de una herramienta de gestión (Jira) los avances y las tareas desarrolladas. También existía el rol de *Application Manager* (APM), el cual era el encargado y responsable de todos los proyectos que se desarrollan con un cliente.

Para el banco Icol, el APM (director inicial de la presente práctica), en compañía de un grupo de trabajo que contaba con experiencia previa en proyectos con el cliente, ubicaban los practicantes de acuerdo con sus perfiles y las necesidades que existieran en el momento. También están encargados de asignar funciones o reubicarlos en caso de ser necesario.

En ese momento se continuaba trabajando en la transformación digital de algunos procesos relacionados con la comunicación entre el cliente y la organización. Debido a la pandemia y la situación de no presencialidad, se debió acelerar esta gestión de procesos. Uno de ellos era la evidencia de trabajo y resultados, para lo cual se decidió en su totalidad por parte del cliente, el uso de Jira, herramienta que permite visualizar las actividades que estaba realizando cada persona, cuánto tiempo le había tomado y la descripción del requerimiento. El cliente solicitaba estrictamente el uso de esta herramienta, para lo cual se hicieron variadas capacitaciones con todas las personas que estuvieran involucradas en el proyecto, ya que era tomado en cuenta en el proceso de facturación y los informes generados por la herramienta debían ser claros.

2.1.4 Tecnologías

En cuanto a las tecnologías utilizadas en el proyecto se evidencia el uso de *frameworks* y entornos de desarrollo de amplio uso. El proyecto cuenta con desarrollo nativo, para poder disponer el móvil del usuario según sea necesario, pero también,

algunos subproyectos son desarrollados en *frameworks* muy populares en la actualidad como Angular, para el desarrollo de las WebView que contiene la aplicación para algunos flujos de transacciones, y por otro lado, Spring para el desarrollo backend con arquitectura de microservicios.

Los programadores que hacen parte del proyecto son los encargados de desarrollar para el sistema operativo Android o iOS de manera nativa utilizando el lenguaje de programación Java o Swift, respectivamente. Esto hace posible ofrecer soporte casi para cualquier dispositivo de la actualidad, según las necesidades que presente la aplicación.

Dentro del rol desempeñado en el desarrollo frontend, se utilizaron diferentes bibliotecas, técnicas y prácticas, que permitían organizar adecuadamente el proyecto, de acuerdo a las directrices de programadores con más experiencia. Por ejemplo, fue necesario utilizar Mocks que proporciona el framework Jest, añadido como una biblioteca de *testing*. Cada mock simula una respuesta a una petición al backend permitiendo obtener una respuesta acertada o no acertada que será interpretada por la aplicación. Los mocks se utilizan para poder realizar las pruebas unitarias y de integración, y para probar ciertos requisitos del frontend aún sin contar con los despliegues de los servidores del backend.

Para el flujo de las transacciones en las aplicaciones se usaron WebView. Cada uno de los flujos correspondía a un subproyecto, gestionado por un equipo diferente. En todas las WebView se usó una biblioteca de estilos desarrollada por un equipo de otro país. Es importante señalar que algunos desarrolladores manifestaron que existían algunos errores en la traducción de los documentos técnicos asociados a esta biblioteca.

Por parte del back-end, en el corto tiempo de capacitación, se pudo observar el uso del patrón hexagonal con algunas adaptaciones que había realizado el líder técnico anterior del subproyecto, que permitía que cada microservicio fuera recibido, validado y enviado a la gestión interna del cliente de la manera más rápida y óptima, utilizando bien sea API REST o SOAP para la comunicación.

En síntesis, se puede concluir que las tecnologías, frameworks, lenguajes de programación y bibliotecas utilizadas, fueron aptas para cubrir las necesidades del cliente, permitiendo abarcar temas como la eficiencia desde el desarrollo nativo, la practicidad con los modelos de diseño previamente creados de las WebView y el uso de herramientas que se encuentran a la vanguardia actualmente.

2.1.5 Aplicación de metodologías ágiles

Dentro del proyecto se aplican algunos componentes de Scrum, sin adoptarla en su totalidad. Desde las funciones ejercidas en el rol asignado, en paralelo al trabajo

realizado diariamente se logró entrevistar a algunos miembros del equipo con los que se trabajó en el área frontend, para el subproyecto en curso. A dos personas del equipo que llevaban más tiempo en la empresa se les realizó la siguiente pregunta: ¿Cómo evalúa usted la aplicación de las metodologías ágiles en el proyecto? A continuación, se presenta un extracto de las respuestas obtenidas:

- Senior Front-end (Angular): "...Las metodologías ágiles son utilizadas como base, pero no se siguen en su totalidad ya que en el sector bancario aún se tienen procesos retrógrados que afectan los tiempos y su intento por reducirlos. Accenture puede aplicar bien las metodologías, pero si el cliente (banco) no aporta lo que debe, existirán ciertas falencias." Aporta también que según el cliente, el uso de las metodologías ágiles llevaría consigo más tiempo y dinero.

- Mid Front-end (Angular): "...La aplicación de la metodología tiene ciertos inconvenientes ya que el cliente no está comprometido en su totalidad involucrándose como se necesita".

En el tiempo que se pertenecía al subproyecto con este cliente, se pudo observar la forma como se ponía en práctica desde el rol ejercido (parte de un equipo de desarrollo) esta metodología, realizando las siguientes actividades:

- Reuniones diarias de seguimiento.

Las reuniones de seguimiento tomaban alrededor de 30 minutos al día. Cada persona exponía el trabajo realizado, los problemas que ha tenido y los problemas por resolver. En algunas ocasiones la reunión virtual tomaba más del tiempo mencionado y perdía su rumbo, pero generalmente se tenía conciencia de ello y se trataba de controlar. En estas reuniones debía estar todo el equipo de desarrollo y el líder técnico. En ocasiones se contaba con la asistencia del scrum master.

- Intervenciones del Scrum Master (SM)

Esta persona era la encargada de controlar el tiempo de las reuniones y que fueran totalmente objetivas, además de ello, en algunas ocasiones hablaba con el cliente para poder organizar los sprint y gestionar el backlog, apoyado por la herramienta de gestión Jira, conocida por el SM y avalada por el cliente. Tenía conocimiento de la mayoría de subproyectos y se encargaba de gestionar las entregas continuas y los reportes de progreso.

- Despliegues continuos utilizando la herramienta Jenkins

Jenkins permitía hacer despliegues continuos, buscando aportar a la calidad del producto. A pesar de no haber utilizado la herramienta, fue posible reconocer que estaba configurada para admitir código fuente y despliegues, siempre y cuando los cambios existentes hayan estado respaldados por pruebas unitarias y de integración. Una vez fuese admitido, se enviaba al equipo de QA donde se continuaba con otros procesos.

- Planificación y revisión de sprint:

Esta actividad era realizada por el scrum master, el líder técnico y en ocasiones, asistía un encargado del cliente. No existió una partición activa dentro de esas actividades.

Se puede concluir entonces que conceptualmente algunos fundamentos de las metodologías ágiles se implementan, tratando de tener intervenciones con el cliente, gestionando entregables, creando listas de actividades que aporten valor, y organizando reuniones de seguimiento permanentes del equipo pero esporádicas con el cliente. Sin embargo, como se puede reconocer, estas son actividades que en su mayoría gestiona Accenture, y no se cuenta con la intervención del cliente.

2.1.6 Aplicación de TDD

Una práctica utilizada por Accenture es Test Driven Development (TDD). A pesar de su potencial, trae consigo algunos retos, como la necesidad de invertir más de tiempo en el desarrollo y de una mayor habilidad en las tecnologías por parte de los desarrolladores. Esto lleva consigo una amplia curva de aprendizaje, que debe ser considerada en programadores con cualquier nivel de experiencia, en especial cuando es la primera vez que se va a aplicar en un proyecto.

En un principio, dado que los practicantes ingresan con roles Junior, o personas sin experiencia, son supervisados por ingenieros Senior con más experiencia. Ellos son los encargados de orientar acerca de las estrategias para desarrollar las diferentes funcionalidades de acuerdo con la arquitectura del sistema. Una orientación muy importante dada a conocer consistía en que se deben programar todas las funcionalidades con sus respectivas pruebas unitarias y de integración, las cuales eran implementadas y ejecutadas antes de realizar un entregable.

Otra apreciación adquirida gracias a las orientaciones que se obtuvieron se centraba en las pruebas a realizar, exponiendo la pregunta ¿Se conoce con exactitud qué funcionalidad es la que se va a probar? Una vez teniendo clara la respuesta a esta pregunta, de acuerdo con los pasos de TDD, se haría en primera instancia que la prueba falle y empezar el desarrollo iterativo que propone la práctica para obtener la prueba unitaria, con el requerimiento finalmente resuelto.

Teniendo en cuenta el hecho de que una persona estuvo monitoreando los avances realizados, se puede decir que se aplicó TDD para algunas de las funcionalidades solicitadas, ya que, no siempre se utilizaba para todos los requerimientos por cuestiones de tiempo. Sobre la ruta no crítica del subproyecto, no se utilizaba esta práctica, simplemente se desarrollaba el requerimiento, y se realizaban o ajustaban las pruebas.

Finalmente, es posible concluir según lo observado en los meses iniciales de práctica, que esta práctica de desarrollo es una buena elección, siempre y cuando se tenga el tiempo suficiente para que las personas sin experiencia aprendan técnicamente a desarrollar las pruebas como se plantea y las personas con más experiencia estén atentas a apoyarlos, aportando calidad al producto sin afectar los tiempos de desarrollo.

La vinculación a los proyectos con el banco Icol terminó cuando el subproyecto al que se pertenecía entrara en su fase final de entrega, para lo cual no se necesitaba todo el apoyo que se tenía inicialmente. Únicamente continuaron los profesionales que trabajaron en él, desde el principio en varios subproyectos.

2.2 Caracterización banco Cubo

El rol desempeñado en este proyecto se enmarca en la gestión de proyectos. Algunas de las funciones principales se centraban en el seguimiento, específicamente, la recolección y presentación de evidencias de los resultados obtenidos y el progreso frente requisitos de software que fueron creados anteriormente. Inicialmente se realizó un reconocimiento para conocer la forma como se estaba llevando a cabo la gestión de proyectos, para posteriormente, llevar a cabo las actividades solicitadas.

2.2.1 Comunicación y tecnologías

Se realizaban dos seguimientos por semana, que involucraban varios proyectos o incidencias con personas influyentes por parte del cliente. La presentación de estos seguimientos se realizaba por cada proyecto o incidencia, apoyados por *dashboards* realizados en Google Data Studio y algunos documentos de Google Sheet. Estas herramientas ofrecen una gran cantidad de servicios que les permite ser muy flexibles, sin embargo, no se considera una solución óptima para algunas necesidades que se presentan en la empresa. A continuación, se describen de forma breve.

- Google Data Studio ofrece diferentes mecanismos de visualización incluyendo gráficos, tablas, filtros y demás información dinámica para el cliente que interactúa con el dashboard. Es posible importar información de diversas fuentes, como Google Analytics, Big Query, hojas de cálculo. Permite al usuario visualizar la información que realmente necesita, con una interfaz amigable. Esta herramienta es muy valiosa, ya que, al conectarse con una fuente de datos, como una base de datos SQL, evita realizar un desarrollo front-end, siempre y cuando se haya estudiado previamente que las necesidades se cubren. Es importante anotar que esta tecnología no permite editar la información presentada.

- Google Sheet es una herramienta de hojas de cálculo que permite realizar diversas operaciones que involucran las celdas, visualizar gráficos, gestionar

filtros, entre otras funciones. Una ventaja de esta herramienta consiste en que se comparte en tiempo real, por lo cual, varias personas pueden interactuar en un documento de forma simultánea. Dado que es una herramienta que puede guardar información con diversos propósitos, es posible tomarla como fuente de datos, permitiendo organizar la información para ser posteriormente analizada, por ejemplo, por un sistema de Inteligencia Artificial.

Dado que el cliente trabaja con la Suite de Google, le interesaba que algunas aplicaciones incluidas fueran utilizadas, por ello, para aportar a la transformación digital de varios proyectos en Accenture se decidió centralizar la presentación de resultados a una sola herramienta, Google Data Studio. De esta forma, a partir de ese momento era necesario normalizar los dashboards existentes con un diseño adecuado para el cliente y que fuera estándar para cualquier informe de seguimiento o resultados.

Para ese momento no se habían estandarizado diseños, los datos no estaban normalizados y podían existir inconsistencias en la muestra de resultados, por esto, ya que Google Data Studio consume los datos, pero el proceso de reconocimiento es relativamente básico (como el reconocimiento de fechas o tipos de números), lo ideal, sería tener una fuente de datos en la cual la información disponible sea confiable.

Usando estas dos herramientas y mediante reuniones semanales, se gestiona la comunicación para el seguimiento de proyectos. En ocasiones se obtenían nuevas incidencias que se ingresaban a la hoja de cálculo y se realizaba una estimación que permitiera llevar registro de su ejecución, desarrollo y finalización. Sin embargo, se reconocía que existía una dificultad en el proceso de la presentación de evidencia, ya que en ocasiones se mostraban avances en las hojas de cálculo, o en dashboard antiguos. Esta situación se complicaba a medida que aparecían nuevas incidencias, que para ese momento, eran más de 50.

2.2.2 Metodologías

En cuanto a las metodologías utilizadas, de acuerdo con el rol ejercido en aquel mes de trabajo donde se encontraba en curso la labor de la gestión de los proyectos, se reconoció la implementación de las metodologías ágiles, acorde al proceso de transformación digital hablado en anteriores ocasiones. Desde el cargo efectuado no se desempeñaba un rol perteneciente a la metodología, ya que se habían planteado unos objetivos a cumplirse, los cuales no implican pertenecer directamente a un equipo, sino participar en el monitoreo de varios proyectos o incidencias en ejecución (ver Figura 1).

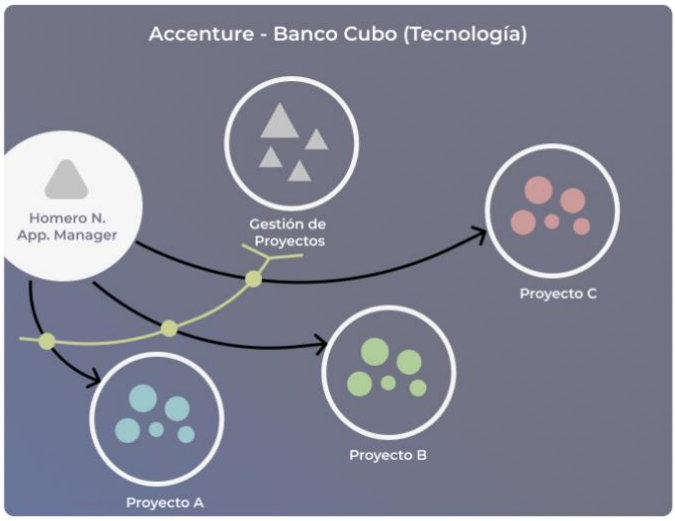


Figura 1: Equipo de tecnología Banco Icol - Accenture. Elaboración propia.

Como se puede observar en la imagen, existían varios grupos que desarrollaban o gestionaban un proyecto o incidencia. Cada uno de ellos cuenta con su líder técnico y se reúne diariamente acorde a las metodologías ágiles, para ejecutar un proyecto. El rol desempeñado fue transversal, y se relacionaba con la gestión de proyectos. El trabajo realizado era enviado directamente al *Application Manager*, quien lideraba en gran parte la comunicación con el cliente.

Algunos equipos que manejaban subproyectos o aplicaciones directamente, se reunían al control de seguimiento diario, y debían mostrar resultados cuando se realizaban, anexo a ello, se llevaban a cabo acompañamientos con el equipo de QA, en los cuales se analizaba el progreso y se hacían las respectivas preguntas sobre las fallas encontradas.

Finalmente, se presentó una propuesta para mejorar el proceso de gestión de la información de cada proyecto o incidencia. Esta propuesta fue aceptada, y se asignó la responsabilidad de implementar la solución que se describe en los siguientes apartados.

2.3 Entorno y viabilidad para la propuesta

Luego de evidenciar las dificultades en la comunicación, la presentación de resultados y la recolección de la información, y de seleccionar la herramienta Google Data Studio, el grupo de proyectos Accenture – Banco Cubo, decidió abrir una convocatoria a pequeña escala para presentar un proyecto que pudiera cubrir esas necesidades. Era necesario crear una herramienta/software que permitiera gestión la información de la muestra de resultados o seguimientos, apoyado con Google Data Studio.

Dado que la mayoría de los profesionales se encontraban en un subproyecto y resolviendo requisitos, era necesario que la herramienta fuese fácil de usar, ya que

era reconocible el problema de que algunas personas actualizaban la información muy rápidamente, sin rectificar lo escrito. Eso resulta vital para la normalización de datos, ya que, si un campo es requerido o tiene un tipo de datos en específico, debe ser llenado correctamente. Este era uno de los problemas más importantes reconocidos por el encargado de Accenture (cliente), ya que habían intentado crear guías o tutoriales para llenar correctamente las hojas de cálculo, pero aun así existían inconsistencias.

Dado que la fuente de datos y la gestión de la información se realizaba a través de Google Sheet, existía una gran cantidad de archivos distribuidos en múltiples carpetas. La cantidad de información que se generaba representaba un problema para los ingenieros, quienes debían navegar sobre la estructura de directorios y ubicar manualmente las hojas de cálculo y los dashboards creados con anterioridad. Esta tarea podría resultar tediosa, haciendo que los encargados de actualizar las fuentes de información realicen su labor momentos antes de las reuniones de seguimiento. Por esta razón, la herramienta además de capturar y almacenar la información que se mostrará en Google Data Studio, debe ser muy intuitiva y fácil de usar, para que reciba total aceptación de los usuarios.

Por parte de las tecnologías y el desarrollo, un requerimiento no funcional muy importante consistía en utilizar mayormente herramientas gratuitas de Google, ya que, generalmente algunos software o servicios en la nube realizan un cobro luego de cierta cantidad de peticiones a sus servidores, sin embargo, los clientes usuales de la aplicación no superarían las 20 personas.

De esta forma, la herramienta aportaría a los procesos de gestión de entrega de evidencias, permitiendo a los ingenieros que actualizan constantemente la información, realizar su labor ágilmente y de manera adecuada, cumpliendo con las solicitudes del cliente.

La apertura de esta propuesta resulta viable e importante ya que en algunas ocasiones se exponen datos inconsistentes en las reuniones de seguimiento con el cliente. Por ello, lo que se busca es presentar información valedera y consistente, solucionando el problema de la recopilación de la información y favoreciendo la comunicación.

Capítulo 3

3. Desarrollo de actividades de gestión de proyectos

A lo largo de la práctica, se desarrollaron diferentes actividades de gestión de proyectos, en su mayoría, sobre el tercer mes de trabajo, el cual fue designado para la participación en un rol que apoyaba diferentes procesos de gestión con el banco Cubo. Inicialmente, cuando se trabajó con el banco Icol, las actividades estaban enmarcadas en su mayoría por desarrollo, reconociendo un proyecto de software a gran escala, y los procesos de gestión de proyectos eran dirigidos por un grupo de personas que conocían todos los proyectos que se estaban realizando con el banco Icol. Ellos eran los encargados de organizar reuniones de control, muestra de resultados, entregables, cronogramas y demás, acorde con las metodologías ágiles, apoyados por un Scrum Master, mientras los desarrolladores se encargaban de llevar a cabo cada funcionalidad solicitada.

Al igual que el banco Icol, por designación de Accenture, se estaban aplicando las metodologías ágiles con el banco Cubo, donde se iba a participar y contribuir en las actividades de gestión de proyectos, las cuales son muy similares a las que realizaba el grupo del banco Icol. Existe una expresión clave que Accenture utiliza para guiar y organizar sus proyectos con diferentes clientes, la cual es: transformación digital, la cual es ampliamente utilizada, sin embargo, la aplicación que se da en este contexto permite cambiar ciertos procesos de gestión apoyándose en nuevas tecnologías para lograr que se automaticen procesos y ocupen menos tiempo haciéndolo más productivo.

Por parte del banco Icol, se realizó un acuerdo con Accenture para dejar de lado las hojas de cálculo, algunas reuniones presenciales o virtuales y documentos extensos que muestran avances o actividades, para trasladar todo a Jira, una herramienta de gestión muy versátil y robusta, la cual estaban dispuestos a pagar. Por esta razón, se encontraban en el proceso de trasladar toda la información de los sprint, las actividades, el *backlog*, los integrantes y demás, que desde el rol de desarrollador podría interesar, y era necesario adaptarse y aprender cómo se iban a manejar los recursos en adelante, a través de capacitaciones, las cuales se estaban realizando con todos los integrantes de los proyectos, sin importar sus roles, para aportar a una transformación digital constante propuesta por Accenture.

3. 1 Actividades de gestión con el banco Cubo

Una vez ocurrido el traslado hacía el banco Cubo, las nuevas responsabilidades estaban enmarcadas en la gestión de proyectos. Es importante recalcar que el rol al cual se pertenecía apoyaba en gran medida las reuniones de gestión, planeación, control y muestra de resultados, sin embargo, este trabajo era realizado como tal por

personas del grupo de trabajo con estudios como administración o ingeniería industrial, incluso practicantes de las mismas áreas, pero a diferencia de ellos, las nuevas responsabilidades estaban dirigidas a apoyar la transformación digital a través de nuevas propuestas de tecnología, que permitieran además de asistir a reuniones o monitorear cronogramas, innovar con soluciones tecnológicas. Por esta razón, se planteó y ejecutó una propuesta para ayudar en ciertas necesidades, la cual será expuesta en el siguiente capítulo.

Inicialmente se realizó un reconocimiento de las actividades a realizar, entre las cuales estaban: asistir periódicamente a reuniones de planeación y control, gestionar herramientas como Google Sheet y Google Data Studio, monitorear las incidencias que se encontraban en diferentes hojas de cálculo, gestionar la presentación de avances, entre otras.

Luego de haber reconocido en gran parte las actividades de trabajo a realizar, se elaboró un plan de trabajo a corto plazo, el cual incluía:

- Organizar la información y recolección de datos por medio de la herramienta Google Sheet.
- Crear y gestionar los dashboards existentes o pendientes, apoyados de la herramienta Google Data Studio.
- Asistir, gestionar y apoyar reuniones de control, seguimiento y presentación de resultados cuando sea necesario.

Toda la información de los proyectos e incidencias se encontraba almacenada en varios archivos de Google Sheet. Existían dos principales los cuales se revisaban con los clientes semanalmente; la inicial tarea fue organizarlos, explotando al máximo algunas funcionalidades de Google Sheet para tratar de que los datos fueran lo más consistentes posibles. Algunos otros archivos ubicados en diferentes carpetas de Google Drive no se utilizaban constantemente y eran principalmente de apoyo o respaldo.

Dado a que los archivos tanto de Google Sheet y Google Data Studio se encontraban en una carpeta compartida con miles de archivos más, se realizó como propuesta de organización una página web en Google Sites, la cual fue desplegada con restricción de acceso a los usuarios “cubo.com” y contenía enlaces a las hojas de cálculo más importantes y a los dashboard realizados anteriormente. Esto permitía orientar a los integrantes de los equipos y dar orden a los documentos, ya que, en vez de buscar en Google Drive, se accedía directamente mediante el enlace a un archivo, y permitía tener su última versión, en caso de haber realizado un respaldo o actualización de ubicación.

Figura 2: Ejemplo de página web creada con Google Sites. Elaboración propia.

Anexo a ello, se mostraba en la página una visualización previa de los dashboard, que era posible abrir y ejecutarse, para mostrar desde ahí los avances en caso de ser necesario.

Respecto a las hojas de cálculo que se utilizaban frecuentemente, fue necesario adaptarlas y optimizarlas utilizando etiquetas y validaciones de campos, para tratar que los ingenieros llenaran la información lo más consistente posible, y en caso de que el cliente tuviera que acceder a la información, se mostrase verídica. Sumado a ello, se cumplió el objetivo de que los dashboards creados anteriormente tuvieran una fuente de información más precisa. A pesar de que ya se había tomado la decisión de realizar la presentación mediante Google Data Studio, no se había avanzado en este cambio, ya que en ocasiones se mostraban al cliente las hojas de cálculo o dashboards creados sin ningún estándar visual. La recolección de datos era un problema y dificultaba mostrar correctamente los avances que se habían realizado en la semana.

Paralelo a la optimización de las hojas de cálculo, para tratar de mitigar el problema de la inconsistencia en los datos, se estaba realizando el control y seguimiento a las incidencias, editando según las reuniones de presentación de avances, filas de datos sobre las hojas de cálculo, para también reconocer la información que ahí se encontraba. Fue necesario reunirse con el equipo de trabajo de Accenture y el banco Cubo para mostrar la nueva configuración de las hojas de cálculo, que incluían algunas mejoras, como se mencionó anteriormente.

Mientras algunas personas también encargadas de la gestión de proyectos levantaban actas y monitoreaban cronogramas, las tareas asignadas realizadas

estaban enfocadas a tratar de cambiar y optimizar algunos procesos de gestión enfocados a la presentación de resultados. Entonces, se estaba buscando la manera de no generar informes que podían ocupar mucho tiempo, sino utilizar una herramienta interactiva que permitiera optimizar y agilizar el proceso. Por esta razón, fue tomado Google Data Studio como herramienta a utilizar para explotar sus funcionalidades y brindar al cliente una mejor forma para evidenciar los avances y llevar el seguimiento.

Anteriormente se habían creado algunos dashboards utilizando Google Data Studio, sin embargo, no contaban con algún diseño o estándar que se mostrase organizado. Por este motivo, la primera labor consistió en reconocer estos dashboard y observar las características adecuadas y aquellas que se podían mejorar. Gracias a esto, se logró una familiarización con la herramienta y fue más fácil reconocer la forma como se estaban realizando las conexiones por medio de Google Sheet actuando como fuente de datos. Se realizó una validación de los tipos de datos y se trabajó en un diseño fácil de usar, intuitivo y apoyado en algunos criterios básicos de la usabilidad, vistos en anteriores proyectos y a lo largo de la carrera. La finalidad de realizar el diseño consistía en crear una plantilla con elementos, colores, y posiciones en los cuales estarían las tablas, listas desplegables, *sliders* o cualquier otro *widget* disponible en la herramienta.

Una vez finalizada y aceptada la propuesta de la plantilla por el asesor de la práctica, que además es el principal usuario del resultado final, se dispuso a rediseñar los anteriores dashboard con las mejoras correspondientes, y crear otros que se encontraban pendientes. Finalmente, se trabajó sobre cuatro dashboard que eran los necesarios, ya que se buscaba renovar los anteriores con la plantilla establecida, y crear los faltantes, para que todos estos informes de evidencia de trabajo se presentaran organizados, estéticos y lo más importante, con información consistente.

Con el fin de evitar almacenar demasiada información en muchas hojas de cálculo y posibles dashboard, se decidió optimizar y comprimir toda la información a dos dashboards, los cuales tenían como fuente de datos las dos hojas de cálculo más importantes que se estaban manejando.

Los cuatro dashboard realizados cuentan con un patrón de diseño el cual permite al usuario tener interfaces semejantes, con las herramientas, logos y widgets que el programa ofrece en ubicaciones muy similares. También se configuraron las tablas y gráficas, para que operen de igual forma, sumando puntos a la usabilidad del sistema. Es importante resaltar que la labor realizada fue esencial desde el rol de la ingeniería de sistemas, ya que a pesar de que en el grupo trabajaban ingenieros industriales, ingenieros en administración y demás, una persona que reconoce sobresalientemente estas herramientas, que tiene conocimientos en programación, usabilidad, fuentes de datos y demás, favorece a que los tiempos de realización de este tipo de requerimientos se acorten.

Figura 3: Ejemplo de Dashboard creado con Google Data Studio. Elaboración propia.

Se realizaron dashboard similares al expuesto anteriormente, aplicando el mismo patrón de diseño, teniendo en cuenta que las herramientas de control, como listas desplegables o entradas de texto, y gráficos o tablas, eran ubicados según la necesidad, el volumen de datos, relevancia, entre otros criterios. Semejante a la figura 3, en la parte superior se ubican los logos de las compañías junto con el título del dashboard, debajo se tenía acceso a todas las herramientas de control que permiten filtrar la información de las tablas o gráficos. En la sección derecha se encontraban algunos datos de menor relevancia, utilizado para estadísticas concretas. Finalmente, inferior a los controles, se encontraban todas las tablas o gráficos interactivos que mostraban los proyectos o incidencias, avances, estados, encargados, descripciones y demás información, que de acuerdo con las reuniones que se tenían con el cliente, se iban identificando.

Figura 4: Patrón de diseño para Dashboard. Elaboración propia.

Gracias a esta organización, los clientes podían interactuar fácilmente con el dashboard, ya que, por ejemplo, sabían dónde encontrar las herramientas de control, ubicadas siempre en la misma posición. Aunque apoyarse en los anteriores dashboard construidos ayudó considerablemente, el cambio en el diseño fue fundamental, ya que se habían incluido los colores corporativos, estandarizado los *layout* y demás detalles que permitían que los informes de avances fueran óptimos, apoyando la transformación digital propuesta por Accenture.

Las reuniones a las que se asistía resultaron muy provechosas para conocer paulatinamente las incidencias o proyectos, y lograr identificar campos o información que se podría agregar y así, optimizar los dashboards, entregando permanentemente valor al cliente.

En relación con otras actividades, como documentos de seguimientos, actas de reuniones, ajustes de cronogramas y todas aquellas actividades de gestión de proyectos, se realizó el apoyo pertinente cuando se requería, ya que como se mencionó anteriormente, existían ingenieros o practicantes que laboraban específicamente sobre el tema. Por ello, mientras estas personas ingresaban, editaban o borraban la información, por medio de las responsabilidades adquiridas se buscaba aportar a diferentes procesos para que fueran más rápidos y eficientes, a través de herramientas informáticas.

3.2 Actividades de gestión sobre la propuesta planteada

De acuerdo con la convocatoria a pequeña escala realizada entre Accenture y el Banco Cubo, la propuesta planteada debía presentarse debidamente organizada, por esta razón, se tomó de guía Scrum como marco de trabajo, y de esta forma, organizar el proyecto acorde con el interés de Accenture por aplicar las metodologías ágiles. Debido a que consistía en una convocatoria interna, no era necesario crear actas de inicio o constitución del proyecto, pero sí un documento corto, apoyado por la sustentación de la propuesta, en el cual se debía describir cómo se iba a realizar el software, que tecnologías se usarían, cuales eran los requisitos, el alcance identificado y un cronograma, posterior a la aceptación.

Inicialmente, cuando se abrió la convocatoria se realizaron 2 mockups como guía para empezar lo antes posible a desarrollar un demo funcional, que permitiese visualizar cómo iba a funcionar el software, ya que se tenían menos de 2 semanas para realizar una propuesta y poder ser ejecutada. La convocatoria se abrió entre los diferentes practicantes que se encontraban trabajando con el banco cubo, alrededor de 5 personas que debían exponer cómo solucionar la problemática de comunicación, tecnologías y recolección de información.

Posterior a realizar el demo y exponer la solución, la propuesta expuesta fue elegida para ejecutarse, por ende, se debía empezar a estructurar el proyecto que tenía de plazo máximo dos meses, que coincidía con el final de la práctica profesional. A partir de ese momento, se ocuparía entonces el rol de director/líder del proyecto como principal responsable de que todo lo planeado se cumpliera de manera oportuna. El tiempo con el que se disponía era relativamente corto y no se contaba con un equipo de trabajo permanente, por el contrario, con personas o practicantes que, si tenían disponibilidad de tiempo aparte de sus actividades, se podría contar con ellos, sin embargo, no se tenía contemplada la posibilidad, ya que todos los roles tenían responsabilidades muy exigentes.

3.2.1 Requisitos

De forma paralela al inicio del proyecto en desarrollo, se debía finalizar y aprobar el documento que exponía el plan de trabajo, el cual no debía ser demasiado elaborado, pero debía mostrar la construcción y composición del software. En primera instancia se realizó la extracción de los requisitos funcionales, los cuales inicialmente eran:

- Iniciar sesión en la aplicación únicamente usuarios “cubo.com”
- Crear, editar, eliminar y visualizar formularios con campos dinámicos, cuyos valores puedan ser textos, lista de textos, listas desplegables con valores definidos por el usuario, números y fechas.

- Permitir visualizar en forma de tabla los valores de los formularios, en donde cada valor sea editable respecto a su tipo.
- Agregar filas a un formulario, realizando la validación de las entradas de acuerdo con sus tipos.
- Visualizar u ocultar columnas de un formulario.
- Exportar e importar valores de una hoja de cálculo de Google Sheet.

Los requisitos planteados serían los iniciales, ya que de acuerdo a las metodologías ágiles, existe la posibilidad de flexibilizar el cambio o la agregación de algunos de ellos, intentando no afectar el alcance del proyecto. Por otra parte, algunos requisitos no funcionales eran:

- Es imperativo que el sistema debe ser fácil de usar e intuitivo, para que los usuarios puedan editar la información rápidamente.
- Inicialmente, no se cuenta con un presupuesto.
- El tiempo de ejecución del proyecto es de dos meses como máximo.
- Se deben aplicar las metodologías ágiles, realizando entregas de avance cada semana. Los sprint durarán 1 semana.
- La información debe ser almacenada en una base de datos gratuita.
- Ningún usuario que no tenga el dominio del banco debe poder acceder a alguna información de la aplicación.
- El manual de usuario estará incorporado en la aplicación.
- El sistema debe estar disponible en cualquier momento del día.
- Entregar una aplicación escalable, que pueda crecer sin necesidad de realizar muchos cambios.
- Todo el registro de trabajo debe quedar guardado en la plataforma Jira

Cada uno de los requerimientos anteriormente mencionados fueron expuestos en el documento como requisitos iniciales del sistema. En la ejecución del proyecto, fueron agregados algunos requisitos funcionales que obedecían al alcance del proyecto, ya que algunas librerías facilitan su inclusión. Sin embargo, fueron requeridas algunas horas de más para la ejecución del proyecto. Algunos requisitos adicionados fueron los siguientes:

- Filtrar datos de la tabla
- Incluir paginación en la tabla
- Clonar un formulario
- Mover una fila de datos a otro formulario

Luego de reconocer los requisitos, se realizaron las historias de usuario, las cuales eran en total 16 inicialmente, con sus respectivos criterios de aceptación, relacionadas con 4 historias épicas. Algunas de las HU se encontraban ya realizadas por la construcción del demo, que fue pensado desde un inicio para poder adaptarse y convertirse en producto. Cada HU fue estimada, dando como resultado 1.5 meses de

desarrollo como se describe en el informe de trabajo número 3, sin embargo, en el último mes planeado, algunos requerimientos se adicionaron y la entrega final se pospuso para terminar en dos meses exactamente. Las 4 historias épicas estaban destinadas para: gestionar los accesos GAPI, gestionar los formularios creados, gestionar los datos y la visualización de cada formulario, y finalmente gestionar la creación y edición de cada formulario. Cada una de ellas abarca al menos 3 historias de usuario, las cuales fueron ubicadas en Jira para su ejecución y seguimiento.

3.2.2 Alcance

El alcance del producto estaba limitado de la siguiente forma:

El usuario ingresará a una página web donde únicamente podrá iniciar sesión con su usuario “cubo.com” con dominio de Google, sin más opciones. Una vez se haya iniciado sesión, el usuario visualizará una lista de los formularios creados que ofrecerán las funcionalidades para visualizar la información, eliminar y editar. Cuando el usuario crea o edita un formulario, la interfaz será la misma. Para el caso de crear, todos los campos estarán vacíos, y para el caso de editar, los campos estarán llenos, de acuerdo con la información guardada por el cliente. En esta interfaz, el usuario registrará el nombre, el Google Sheet ID y cada campo del formulario, que tendrá su nombre y tipo.

Cuando el formulario se encuentre creado y se desee agregar información, se llevará a una interfaz que mostrará en forma de tabla todos los datos ingresados anteriormente. El usuario podrá agregar datos a través de un diálogo modal del formulario creado por el usuario. Los datos serán editables desde la tabla desplegada y se permitirá ocultar y mostrar ciertas columnas. Si el usuario lo desea, puede importar o exportar los datos a una hoja de cálculo de Google, que será el puente para llevar la información a los dashboard de Google Data Studio.

Se podría concluir entonces que el software estaba compuesto por dos módulos, el primero de ellos, encargado del usuario, en donde se gestiona el acceso por medio de las cuentas, la interfaz del login y todo lo relacionado. Por otro lado, estaba el módulo más grande e importante que es relativo a la gestión de la información, la recolección y muestra de datos. De esta forma estaban definidos inicialmente los requisitos y el alcance del proyecto que por medio de la herramienta de gestión Jira, sería monitoreado acorde a los sprints y entregables semanales.

3.2.3 Cronograma

Relacionado con el cronograma del proyecto, se llevaron a cabo los sprint que se presentan en la siguiente tabla:

Sprint	Objetivo	Semanas						
		1	2	3	4	5	6	7
	Inicio del proyecto	*13 de mayo de 2021*						
1	Realizar actividades de gestión de proyectos y robustecer el demo							
2	Desarrollar la vista y lógica relacionada con las tablas de datos.							
3	Conexión del sistema con servicios GAPI							
4	Finalizar GAPI - Finalizar edición/creación de formularios							
5	Finalizar interfaces y login							
	Fin del proyecto	*2 de Julio de 2021*						
6	Aportar usabilidad - Solucionar errores							
7	Desarrollar requerimientos finales - Capacitaciones							

Tabla 1: Cronograma planeado

Como se expone en la tabla anterior, el cronograma se definió con base a los sprint planeados. En apartados posteriores se detallan las diferentes actividades que se planearon y realizaron. Los sprint 6 y 7 se agregaron debido a la definición de nuevos requerimientos, los cuales agregaron dos semanas más para completar los 2 meses de trabajo. El cronograma expuesto en el documento fue aceptado, sin embargo, no contenía los sprint 6 y 7.

3.2.3 Costos

En cuanto a los costos del proyecto, existía una gran limitación la cual consistía en hacer un software completo únicamente con las herramientas de libre acceso que proveía la Suite Google. Firebase a pesar de ser una herramienta que genera facturación y no pertenecer a Suite, tiene un rango de valores gratuito de acuerdo a la cantidad de peticiones que se hacen al servidor, pero para este caso, el número de usuarios que utilizaría el software es limitado, alrededor de 20 usuarios, los cuales no entrarán al tiempo o permanecerán conectados generando más de 50.000 peticiones

al día, que es el tope diario de solicitudes gratis [30]. Por esta razón, Firebase, exactamente Firestore fue utilizada como base de datos acudiendo al requerimiento no funcional sobre el costo y el almacenamiento de la información en el proyecto.

El hosting podría ser también una razón por la cual sea necesario pagar algún valor, pero al crear y conectar una aplicación web con Firebase, es posible obtener un hosting con dominio gratuito, accesible desde cualquier dispositivo. Por último, el servicio GAPI también podría generar facturación, pero ofrece los mismos beneficios que Firebase permitiendo gratuitamente hasta 500 solicitudes por 100 segundos, sin límite de uso diario [31]. Este intervalo de peticiones gratuitas se ajusta adecuadamente para las necesidades del proyecto, cumpliendo con la gratuidad del mismo.

3.2.4 Calidad

La gestión de calidad aplicada al proyecto se realizó de acuerdo con lo solicitado por el cliente, en función a los seguimientos semanales y algunos lineamientos de Accenture. Como producto interno, aspectos como la documentación del proyecto no eran necesarias, ya que, no tenía un presupuesto asignado hasta el momento, por ende, los lineamientos de calidad iban orientados al proceso de construcción y el producto. TDD, la práctica de desarrollo elegida permite realizar pruebas unitarias y de integración antes y en paralelo al desarrollo (por el proceso de factorización), de esta forma, la calidad iba a ser evidenciada con el uso de esta práctica. Acorde a lo aprendido con el desarrollo en el banco Icol, se iban a aplicar todos los pasos para generar un producto con calidad desde la escritura de su código fuente. Como se mencionó inicialmente, el cliente podría establecer pautas de seguimiento de calidad, sin embargo, los únicos ajustes que se sugirieron iban relacionados a la usabilidad, el cual es un factor de calidad del producto final. También se realizaba un monitoreo respecto a buenas prácticas de programación y documentación básica del código conforme a las recomendaciones del lenguaje de programación elegido (JavaScript).

3.2.5 Riesgos

Se identificaron algunos riesgos descritos en el documento entregado, los cuales fueron:

- Dado a la situación actual generada por el Covid-19, si el director del proyecto tuviese alguna complicación médica, el proyecto y todas sus funciones pueden ser aplazadas indefinidamente hasta nuevo aviso.
- El progreso del proyecto se acoge a la disponibilidad de tiempo del director del proyecto, respecto a otras actividades que pueden ser asignadas por Accenture.
- No existe un grupo de trabajo estable que aporte continuamente al proyecto.

- La gratuidad del proyecto se mantiene siempre y cuando Firebase y GAPI mantengan las políticas mencionadas.

Relacionado con el segundo riesgo identificado, el asesor de la práctica, principal cliente del proyecto aseguró apartar el suficiente tiempo y espacio para la ejecución y entrega del producto, asignando al director la mínima cantidad de actividades, mitigando la afectación en cuanto al tiempo. El tercer ítem puede ser tomado como riesgo ya que en presencia de una persona que apoye desde un principio, se tendría algún respaldo para que el proyecto no se pausara totalmente, sin embargo, aún identificado el problema, no fue posible encontrar una solución dada la alta ocupación del grupo de trabajo, no obstante, nunca se presento el riesgo en la ejecución del proyecto.

Se identificaron algunos riesgos que, al presentarse, el proyecto podría retrasarse o replantearse, como el primer y cuarto. La situación generada por la pandemia del covid-19 puede afectar a cualquier persona aún tomando las medidas recomendadas de bioseguridad, por esta razón, dadas las condiciones actuales este riesgo debe estar presente. Por otro lado, el cambio de políticas de las tecnologías a utilizar podría hacer que el sistema incurriera en gastos, tema que para la creación, desarrollo y utilidad del producto no está contemplado, sin embargo, siempre se tuvo pleno conocimiento de esta situación y el proyecto se ejecutó con normalidad. Una vez entregado el producto, en la última exposición se realizó una corta recopilación de los riesgos planteados, concluyendo que el proyecto se ejecutó sin eventualidades y ningún riesgo se presentó.

3.2.6 Interesados

No era indispensable identificar los interesados o las personas que tenían relación con el proyecto, pero se realizó la tarea de forma preventiva, obteniendo lo siguiente:

- Homero Nuñez: Asesor de la práctica profesional y principal usuario de la aplicación. Es el encargado de editar y mostrar la información de los proyectos y las incidencias al banco. Es necesario mantener comunicación constante y flexibilidad ante cualquier petición. Cuenta con el máximo poder e interés por el proyecto.
- Equipos de desarrollo: Usuarios que interactúan con el producto entregado. Es recomendable mantener comunicación constante ya que deben estar completamente satisfechos con el resultado final. Cuenta con un poder intermedio y un interés intermedio-bajo.
- Cliente banco Cubo: Clientes de todos los proyectos Accenture - Banco Cubo. Posibles usuarios de la aplicación. Gestionar la comunicación no es sencillo, pero se puede lograr a través de Homero N. No cuentan con un poder sobresaliente, pero es importante que el producto final sea de su agrado.

- Felipe Pieschacon: Practicante encargado de la gestión de los proyectos con el banco Cubo. Potencial cliente de la aplicación. No cuenta con poder sobresaliente pero sí influye en las decisiones.

Dadas las características de la aplicación, fue necesario estar atento a cada uno de los interesados del proyecto. En varias ocasiones se citó al grupo de trabajo para revisar los avances, y en cada entrega o seguimiento, era indispensable la asistencia tanto de Felipe Pieschacon como de Homero Nuñez.

3.2.7 Seguimiento y finalización de la propuesta

Luego de iniciar el proyecto, se realizaban de uno a dos seguimientos semanales con Felipe Pieschacon y Homero Nuñez, en los cuales se realizaba un monitoreo del avance a través la herramienta de gestión Jira dado que ahí se encontraba el backlog y la planeación de cada Sprint. De acuerdo con los avances mostrados en cada reunión, los requerimientos faltantes y el tiempo programado, se podía modificar el sprint con los requerimientos de la siguiente semana.

Aunque lo ideal en las metodologías ágiles sería realizar un seguimiento diario como equipo y permanente con el cliente, se llegó a un acuerdo con los interesados, que también permanecían al equipo, realizar al menos un seguimiento a la semana de acuerdo a la disponibilidad de tiempo de Homero Nuñez, ya que al ser el *Application Manager* con el banco cubo, las responsabilidades tomaban gran parte de su tiempo. De esta manera las reuniones no tomaban más de 30 minutos para presentar los avances, requisitos y monitoreo del proyecto. Estos acuerdos permitieron desarrollar exitosamente el proyecto ajustándose a las condiciones, finalizando todas las funcionalidades del *backlog* y sprint.

Como último requerimiento con un enfoque técnico, fue realizado el manual de la aplicación el cual se incorporó como otra página web perteneciente al proyecto. Al realizar la entrega del proyecto se revisaron todas las funcionalidades de la aplicación conforme al manual realizado, obteniendo una excelente respuesta del cliente, con una alta satisfacción del producto realizado.

Por cuestiones de tiempo, el aumento que tuvo el proyecto de dos semanas, las condiciones de entrega, los acuerdos internos entre Accenture y el banco cubo, y demás, no fue solicitada un acta de cierre del proyecto, sin embargo, esta condición también se presenta dado a que toda la información del proyecto quedó registrada en Jira y en un repositorio disponible en cualquier momento de la mano de los clientes. También es importante recalcar que el proyecto quedó abierto a cualquier desarrollador que estuviese encargado de agregar módulos a la aplicación, dado que las funciones más importantes que contienen lógica, repositorio de datos o Redux se encuentran debidamente comentadas según las recomendaciones para el lenguaje de programación JavaScript.

Capítulo 4

4. Desarrollo de funcionalidades

El desarrollo de software se realizó principalmente en dos de los tres roles cubiertos en la práctica. Inicialmente se apoyó con los requerimientos del banco Icol, cuyo proceso fue de mucho aprendizaje, comprendiendo cómo se iba a participar en tal proyecto de gran magnitud y reconociendo el grupo de trabajo. Por otro lado, una vez aceptada la propuesta con el banco Cubo y Accenture, se inició el desarrollo según lo planeado en cada sprint, implementando y documentando el total de las funcionalidades de la aplicación “Accenture Forms”. También fue necesario realizar las pruebas, cuyo proceso fue esencial por la práctica de desarrollo seleccionada (TDD).

4.1 Funcionalidades con el banco Icol

Las funcionalidades para apoyar el desarrollo de software fueron iniciadas una vez se tuvo acceso a toda la información de los repositorios. Fue necesario realizar una configuración global del equipo de cómputo y específica en algunos programas internos de la organización y el gestor de paquetes de NodeJS (NPM).

Posteriormente se solicitó un permiso como desarrollador para tener acceso a internet, no en su totalidad por temas de seguridad y restricciones del banco, pero si a páginas que pueden brindar ayudas, como StackOverflow, documentaciones de tecnologías como Angular, e incluso una página que es utilizada como base para los estilos de la aplicación realizada por la misma empresa bancaria, pero en otro país. Siempre era necesario conectarse desde el equipo que Accenture envía a los hogares de sus funcionarios, a otro equipo remoto que se encuentra en la ciudad de Bogotá, en las oficinas del banco.

En primera instancia, no se tenía el acceso a toda la información, ni los programas configurados, únicamente el equipo enviado por la empresa, sin embargo, se comunicó que el apoyo iba a ser requerido en el back-end temporalmente, donde no se obtuvo valiosa información del proyecto en curso, ya que no se había decidido el rol definitivo, de tal forma que fue necesario tomar algunas guías que brinda Accenture, y otras por aparte sobre la tecnología SpringBoot utilizada para el proyecto. En aquel momento se utilizaban microservicios como arquitectura del sistema, con el patrón hexagonal adaptado según los requerimientos, por el anterior líder técnico del grupo de trabajo.

Una vez un practicante que va a ejercer determinado rol demuestra que tiene las bases suficientes para iniciar a desarrollar y apoyar en las tareas, se gestiona el acceso a los repositorios con el líder técnico, sin embargo, al corto tiempo de haber

ingresado, se realizó un cambio de rol debido a la experiencia en otras tecnologías que ya se poseían desde meses anteriores, incluso en ámbitos empresariales. Desde ese momento se realizó la gestión de los repositorios para obtener toda la información necesaria desde el rol de desarrollador frontend para el subproyecto “retiros sin tarjeta”.

Al iniciar en el desarrollo front-end, fue realizada una inducción, dado que se estaban efectuando algunos cambios entre el banco Icol y Accenture, llevado a cabo por el grupo de gestión de proyectos. El cambio consistía en monitorear todos el proyecto y sus subproyectos, desde la herramienta de gestión Jira, la cual es intuitiva en cuanto al manejo, pero considerablemente robusta y era estrictamente necesario ingresar bien los datos para que la información almacenada sea consistente, por esta razón, una de las ingenieras estaba al tanto explicando cómo se debía realizar, por ejemplo, la evidencia de trabajo diaria, ya que debía esta soportada una HU, un error, una petición del equipo de QA, un nuevo requerimiento o tarea, entre otras.

Cuando el rol ya se encontraba definido y se habían obtenido todos los accesos, se podía empezar el desarrollo, el cual iba siempre apoyado por un programador Senior de las tecnologías a utilizar. Los requerimientos se presentaban y distribuían diariamente en el equipo frontend del subproyecto, a través de Jira. Todas las tareas eran reportadas en la herramienta de gestión y posteriormente se iniciaban, este proceso era necesario ya que existía un protocolo ante la aparición de errores, donde era necesario reportar el acontecimiento, el encargado y la estimación de tiempo, todo esto, con relación a lo establecido por el equipo de gestión de proyectos.

En algunas ocasiones fue necesario crear incidencias o tareas, las cuales se hacían en conjunto con el desarrollador experimentado, actuando como encargado del aprendizaje, los resultados y el monitoreo. De acuerdo con los estándares definidos por Accenture, el desarrollo debía ir guiado por pruebas acorde a TDD, sin embargo, por cuestiones de tiempo algunas pruebas se realizan posterior a cumplir el requerimiento, dependiendo de si afectaba o no la ruta crítica que debía seguir el usuario de la aplicación, de tal forma que al iniciar cada funcionalidad se realizaba el siguiente análisis:

- Si el requerimiento afectaba de una u otra manera la ruta crítica del usuario, respecto al llenado de información de un formulario, validación de datos y navegación en las rutas, es indispensable el uso de TDD.
- Si el requerimiento no afecta la ruta crítica, es opcional utilizar TDD.

Según lo observado en la ejecución de las pruebas, a los desarrolladores Junior les resulta más difícil plantear una prueba unitaria que sea óptima, o en su defecto, que sí pruebe la funcionalidad. La experiencia implica conocer a fondo el framework de prueba, en este caso Jest, y la habilidad como tal de probar la solución del requerimiento, que va poco a poco desarrollándose en un programador.

Alrededor de un mes y medio se apoyó en el desarrollo de software para el banco Icol aportando a la solución de todo tipo de requerimientos, que involucraban HTML, hojas de estilo y lógica de negocio desde el frontend, utilizando Angular con Typescript. Algunos ejemplos de las actividades realizadas se listan a continuación:

- Apoyo en pruebas unitarias y de integración para las WebView que involucraron recepción y envío de datos
- Adaptación de las interfaces acorde a los mockups y ajustes para mejorar la usabilidad
- Solucionar errores solicitados por el equipo de QA
- Adaptación de hojas de estilos aportadas por el cliente.
- Para el desarrollo y las pruebas, se realizaban los mocks que simulaban respuestas del backend, los cuales se actualizaban y habilitaban según lo desarrollado.

Todos los cambios realizados, aún estando monitoreados debían subirse a una rama específica utilizando git, a una plataforma de gestión de repositorios, de esta manera, cuando se agruparan ciertos requisitos que estaban relacionados entre sí, se realizara un *Pull Request* que era revisado por un grupo de personas, para ser enviado a equipo de QA. Si los cambios son aprobados, pasan a una rama que controla lo que será desplegado en producción y mostrado en los avances del proyecto.

Finalizando el segundo mes de trabajo y el proyecto estar en una etapa final, algunos requerimientos faltantes acordados por el grupo de gestión de proyectos acorde al alcance general, podrían ser terminados por un grupo de ingenieros selectos que estaban contratados para ejercer labores de desarrollo para este cliente en específico y contaban con años de experiencia en ello. Estas personas serían las encargadas de realizar la entrega final, uniendo cada producto de un subproyecto, y harían parte del mantenimiento que requiere el software. Finalmente, era necesario hacer los llamados a las WebView desde la aplicación de cada sistema operativo, comprobando que todo estuviera funcionando correctamente para exponer la aplicación a las tiendas virtuales como Play Store y App Store.

4.2 Funcionalidades desarrolladas en la propuesta

Para presentar la propuesta, se decidió crear un demo funcional guiándose en mockups básicos creados con anterioridad para dar una idea general de cómo luciría la aplicación. El desarrollo inició en paralelo a las actividades de gestión que se estaban realizando en el momento. Este demo contaría con una idea sobre el contenido de la aplicación y el flujo o ruta principal que seguiría el usuario para utilizar la aplicación.

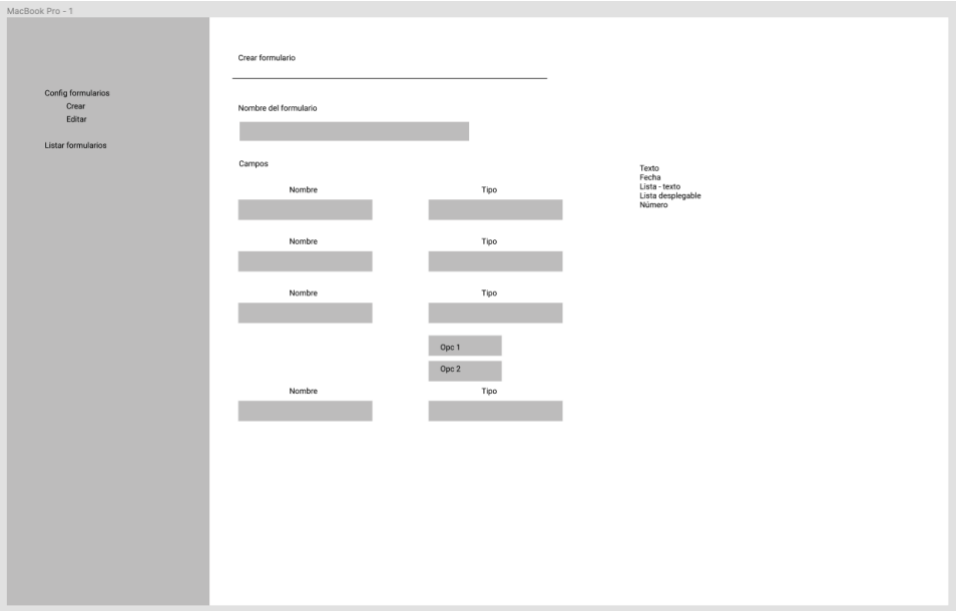


Figura 5: Mockup inicial de vista "crear/editar formulario". Elaboración propia.

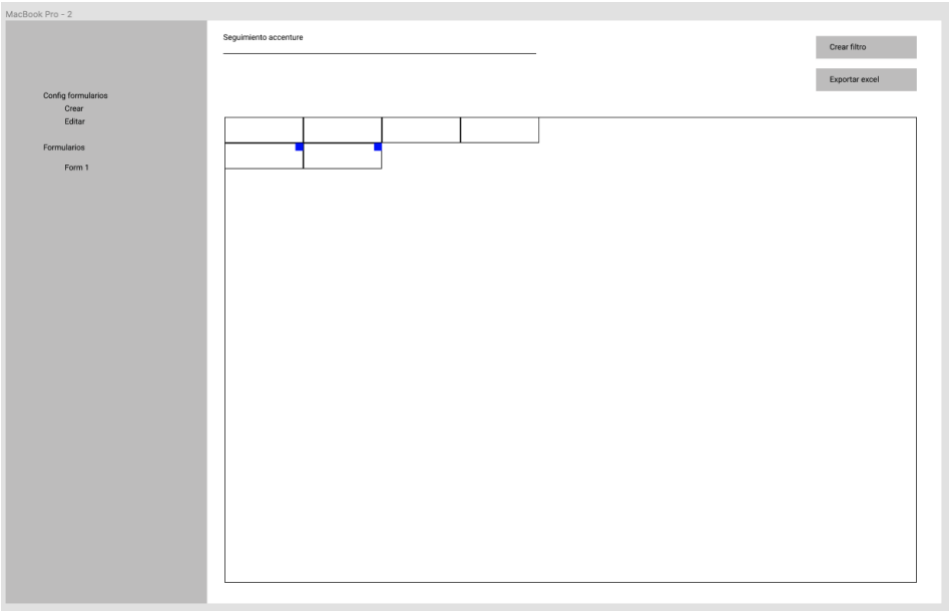


Figura 6: Mockup inicial de vista "editar/visualizar información". Elaboración propia.

La intención principal consistía en mostrar las interfaces principales que iba a tener la aplicación y algunas funcionalidades básicas, como crear un formulario y poder ingresar datos a las filas de la tabla, lo cual correspondía a las interacciones comunes que tendría un usuario con la aplicación. El demo se creó con la idea de ser la base principal para implementar todas las funcionalidades y convertirse en el producto final para ser trabajado hasta terminal la práctica.

Una vez aceptada la propuesta, se empezó a trabajar en paralelo entre las actividades de gestión de proyecto y el desarrollo acorde a la propuesta planteada. En el documento inicial se establecieron los primeros 5 sprints de trabajo.

4.2.1 Sprint 1

En el primer sprint se designó parte del tiempo para terminar algunas actividades de gestión de la propuesta como el documento inicial y las HU del software completo. Cuando se dio inicio, en relación al desarrollo del aplicativo, se tenía una parte funcional del demo, y se planeaba avanzar en las historias épicas relacionadas con la creación de un formulario.

Una vez obtenidas las historias de usuario, estas fueron transcritas al software Jira para poder monitorear los avances, el registro de actividades e informes que provee la herramienta, eso de acuerdo a los requerimientos del cliente. Jira fue configurado acorde a las recomendaciones de Accenture, almacenando el backlog, en donde las estimaciones iniciales indicaban un mes y medio de trabajo.

Respecto a las autorizaciones, fue necesario configurar una cuenta de Google proporcionada por el cliente, donde iban a almacenarse todos los proyectos virtuales de Firebase y GAPI, ya que debía ser una cuenta aparte, que se mantuviera indefinidamente. Fue así como se inició la conexión entre la aplicación local y los proyectos virtuales, agregando determinadas bibliotecas, a través de NPM y ciertos archivos JSON de configuración, de acuerdo con la documentación de la página oficial de Firebase y algunos conocimientos previos adquiridos en proyectos anteriores, dando como resultado una conexión exitosa.

Como gestor de estados de la aplicación y acorde a la arquitectura por capas utilizada para este proyecto, se realizaron las configuraciones iniciales de Redux, herramienta ampliamente utilizada por desarrolladores que utilizan React y sus frameworks, para desarrollar la capa de negocio de la aplicación.

La capa de acceso a datos estaría definida por varias funciones que serían llamadas desde las acciones de cada componente Redux. Se hace referencia a funciones, ya que es la forma más utilizada cuando se trabaja con Javascript, exportando procedimientos, funciones, módulos, objetos y demás en los diferentes archivos “.js”. Las funciones se agrupan de acuerdo a los llamados que se realicen a la base de datos, para este caso, como se va a almacenar información de formularios y sus datos únicamente, en un solo archivo se ubicaron las peticiones, y se exportaba por separado cada función como crear, actualizar, obtener y eliminar, entre algunas otras, simulando tener una clase DAO (Data Access Object).

La capa de negocio de la aplicación se encuentra definida en cada acción de un componente Redux utilizado. Se definieron 3 de ellos: formularios, autenticación y

spreadsheets. Cada una de las acciones contenía la lógica necesaria para recibir la información de la interfaz de usuario (a través de un evento), realizar algún proceso de lógica si es necesario y llamar a la base de datos o a algún endpoint como el de GAPI a través de la capa de acceso a datos. Las acciones podrían realizar un *dispatch* al *reducer* para cambiar el estado de la aplicación de forma reactiva, en caso de ser necesario, por ejemplo, cuando se realiza una petición GET y la información obtenida se desea almacenar globalmente a través del estado de la aplicación.

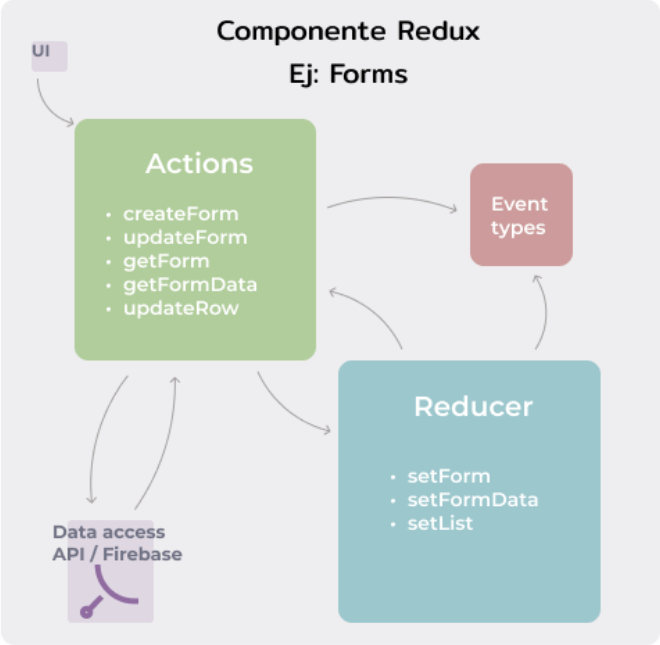


Figura 7: Ejemplo de componente Redux "Forms" creado. Elaboración propia.

La capa de presentación está conformada por componentes que se crean utilizando JSX, que es una extensión de la sintaxis de JavaScript [32]. Algunos de ellos son accesibles desde una ruta, llamados componentes padres, y otros no tienen una ruta, pero son componentes hijos que son llamados desde los padres. En ocasiones pueden desarrollarse para ser reutilizables. Anexo a esto, se utilizaron librerías como Formik, que provee múltiples ayudas para el manejo de estados de formularios, y dado que la herramienta básicamente se componía de formularios, fue de gran ayuda. Por otro lado, para el diseño y estilización de la aplicación, se utilizó SASS, más específicamente SCSS, tecnología que permite controlar las hojas de estilo a un nivel más avanzado, permitiendo la creación de variables, funciones y operadores de ayuda. Bootstrap 5, librería que también facilita el uso de las hojas de estilo, agregando clases en las etiquetas HTML, fue utilizada para brindar estilo a botones, barra de navegación y al *layout*, ahorrando tiempo. Cabe resaltar que desde esta capa se ejecutan los eventos a través de una función *dispatch*, proveída por un *hook* de react-redux, que envía como parámetro la acción que se va a realizar.

La arquitectura por capas en React permite a través de los diferentes componentes Redux, poder realizar la lógica del negocio en las acciones, siempre y cuando no sea

demasiado compleja o costosa en cuanto a recurso computacional, evitando la creación de múltiples controladores en comparación con arquitecturas diferentes. Como la lógica de la aplicación no era tan extensa y costosa, se decidió realizarla sobre las acciones que hacían parte de cada componente Redux, creando una solución rápida y eficiente que se ajustaba a las necesidades.

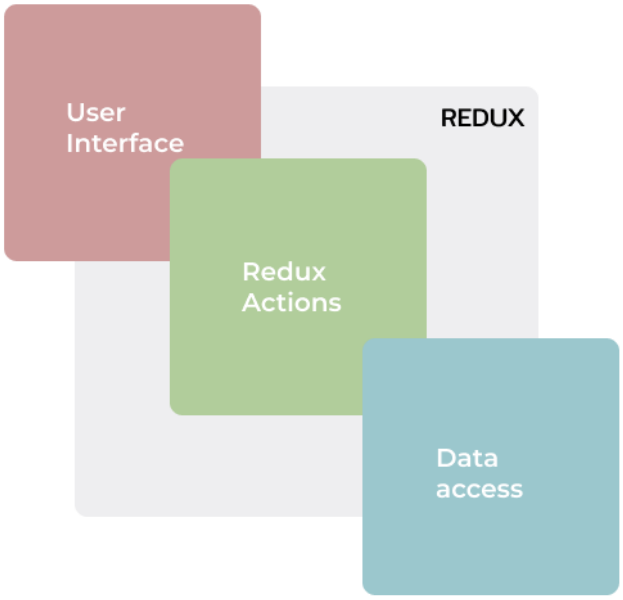


Figura 8: Arquitectura de la aplicación Accenture Forms. Elaboración propia.

Teniendo ya una visión holística de la arquitectura de la aplicación y habiendo realizado las configuraciones de las bases de datos y los componentes Redux, se inició el desarrollo realizando la pantalla de crear un formulario, junto con las primeras funciones del componente redux “forms”, que se conectaba a la base de datos permitiendo guardar los campos del formulario dinámico, como su nombre, campos y sus tipos, y el identificador de Google Sheet.

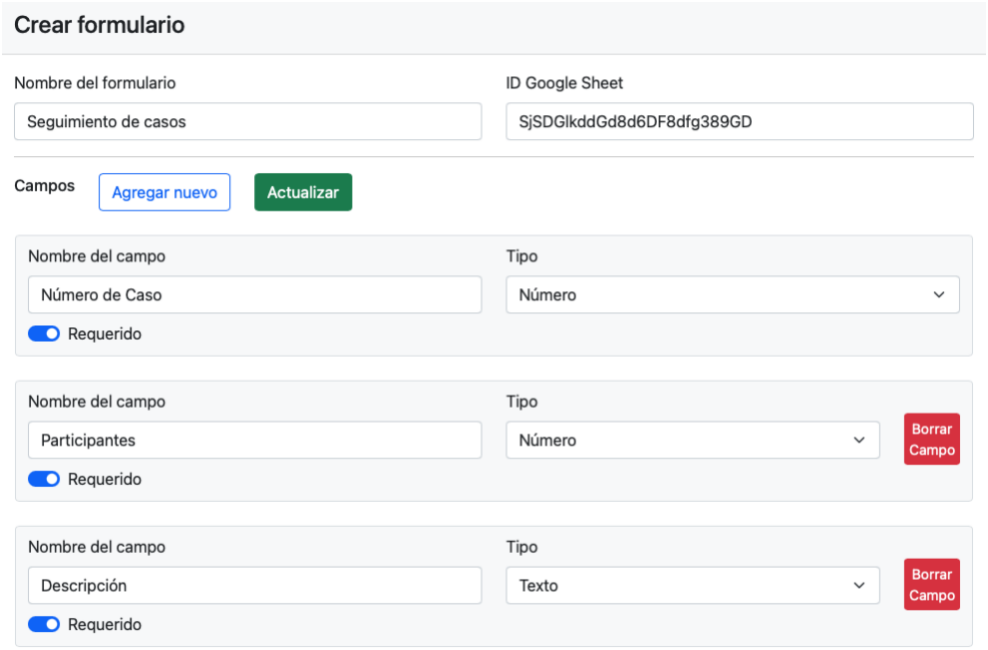


Figura 9: Vista "crear/editar formulario" con datos de ejemplo. Elaboración propia.

Inicialmente, estos eran los campos que se encontraban en la vista, siendo todos *strings* obligatorios. El objetivo de realizar esta vista como primer paso, es poder mostrar al usuario un formulario donde se ingresen los campos que él desee para poder crear un formulario dinámico y de esta forma, que en las próximas pantallas a construirse, los usuarios ingresen datos al mismo. Como se puede observar en la imagen anterior, en esta pantalla se puede ingresar, por ejemplo, lo siguiente: como nombre del formulario "Seguimiento de casos", y en los campos puede ingresarse "Número de Caso" de tipo "número", "Participantes" de tipo "número" y "Descripción" de tipo "texto", para finalmente crear un formulario con esos campos y sea posible ingresar valores.

Formik, la librería que ayuda al control de estados de formularios permite realizar la validación de los valores que el usuario ha ingresado a través de esquemas, que son proveídos por otras librerías, como Yup o Joy. Se utilizó Yup, ya que es una recomendación de la documentación oficial de Formik, lo que facilitó en gran medida la validación de los datos de entrada. En un principio, el formulario perteneciente a la vista de "crear/editar formulario" (Figura 9) llevó un tiempo y esfuerzo considerable, ya que era necesario validar datos dinámicamente, siendo una parte fundamental en cuanto al desarrollo del primer sprint. El formulario ya se había creado junto con la vista (o componente padre) en el demo, pero ahora, era necesario finalizarlo, quitando los campos predeterminados y aplicando la lógica del negocio.

Existieron algunos cambios en el *layout* de la aplicación de acuerdo con los mockups, para favorecer la usabilidad, ya que, por ejemplo, tener una barra lateral, aún siendo removible, ocuparía espacio a lo ancho de la pantalla, quitando visibilidad a lo realmente importante que es la tabla de datos. En algunas ocasiones estas tablas podrían tener más de 12 columnas, de esta manera, resultaba mejor apartar todo el

ancho disponible, por esto, se decidió cambiar la barra de navegación del lado izquierdo al apartado superior, donde gracias a Bootstrap es fácil de crear y ubicar esta barra de navegación. Otra razón para realizar este cambio es que no se iban a crear una cantidad considerable de enlaces de navegación hacía pantallas de la aplicación, lo que resultaba mejor en una barra superior que en una barra lateral, la cual generalmente se utiliza cuando la aplicación permite navegar entre múltiples pantallas.

A medida que se avanzaba en el desarrollo de esta vista y su funcionalidad, se iban realizando las pruebas como lo indica TDD, ya que se aplicó el mismo concepto que se utilizó para el banco Icol, relacionado a la ruta crítica de la aplicación. La creación de los formularios fue considerada ruta crítica, por ello, las pruebas iban de la mano con el desarrollo de las funcionalidades. Dado que fue el primer sprint y la primera vez que se aplicaba sin un grupo de trabajo o una persona experimentada verificando lo realizado, tomó cierto tiempo acostumbrarse a esta modalidad de desarrollo, pero aún así se continuó aplicándola, para finalmente cumplir el objetivo sprint, que consistía en entregar la creación de formularios, lo cual fue expuesto al cliente, quedando satisfecho con los avances.

4.2.2 Sprint 2

De acuerdo con la planeación inicial, el sprint 2 estaría enfocado a la visualización y edición de los datos de un formulario creado por el usuario, mostrados a través de una tabla, similar a Google Sheet, aplicando los requisitos acordados.

Una vez creado el formulario con los campos y características que se muestran en la vista trabajada en el anterior sprint, fue necesario guardar toda esta información en la base de datos no relacional seleccionada para este proyecto, lo cual también se finalizó en el anterior sprint. Dado que solo existían dos colecciones “form” y “data”, y esta última dependía del formulario creado (form), la base de datos tenía la siguiente estructura:

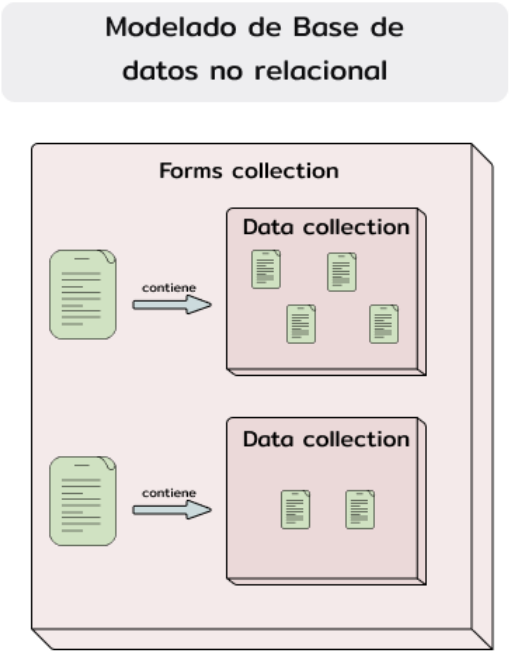


Figura 10: Modelado de base de datos no relacional. Elaboración propia.

Cada formulario que creaba el usuario era guardado como un documento en la colección “forms”, donde se almacenaba cada campo, su tipo y demás, permitiendo que en la vista que sería trabajada en el presente sprint, se obtenga la información (data), donde el usuario agregará nuevas filas de datos a través de un formulario que el mismo usuario habría creado, a inmediatamente poder visualizarlo y editarlo, a través de las celdas.

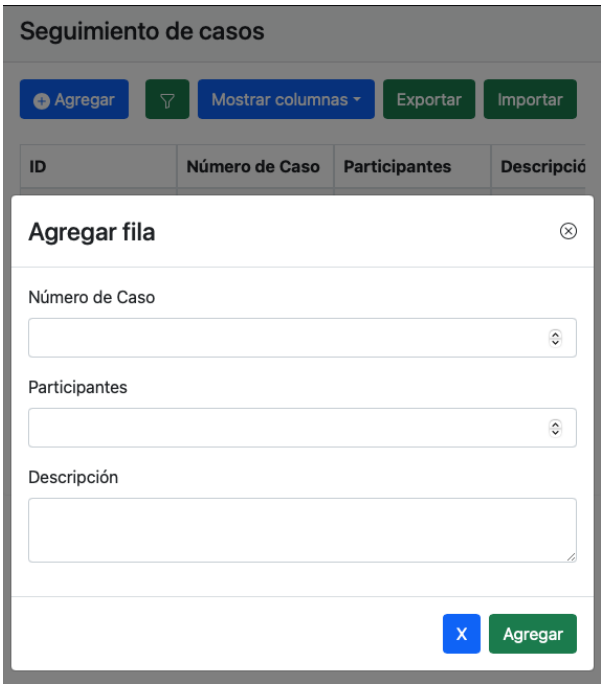


Figura 11: Formulario creado por el usuario sobre un diálogo modal en la vista “editar/visualizar información”. Elaboración propia.

Como se puede observar en la imagen anterior, el formulario que el usuario había creado se muestra como un diálogo modal para poder agregar una fila de datos. Este modal es abierto cuando el usuario presiona el botón de “+ Agregar” que se observa por detrás del diálogo, en la esquina superior izquierda. Cuando se presiona el botón de “Agregar”, se ejecuta un evento que llama una acción para guardar esta nueva fila de datos en un nuevo documento de la colección “data” del formulario accedido, bajo la ruta:

```
“forms/{id-form}/data/{optional-id}”
```

La ruta representa las colecciones y documentos en donde se almacenará la información.

Cabe resaltar que esta interacción relacionada con el ingreso de una nueva fila, es considerada ruta crítica de la aplicación, por ende, se realizó el desarrollo guiado por pruebas, sin embargo, gracias a la librería Formik, el manejo de los formularios estaba casi en su totalidad probado por sus colaboradores, solo había que probar que la información se guardará y se mostrará satisfactoriamente.

De esta forma, en el presente sprint se trabajó en la muestra correcta del modal de acuerdo al formulario que el usuario había creado, con sus respectivos campos y tipos, y poder almacenar la información que un usuario ingrese al formulario en la base de datos, para posteriormente mostrarlo en la tabla de datos. Una parte primordial del sprint consistía en hacer editable el contenido de la tabla, además de visualizar los datos.

Para renderizar la tabla de datos se utilizó la librería “react-table”, la cual brinda múltiples ayudas para el manejo de tablas en React, como la paginación, filtrado de datos, contenido editable y demás. A través de la documentación, se realizó la funcionalidad del contenido editable, ya que en cada celda se podía conocer cuál era el documento de la base de datos que se iba a editar, su posición y nuevo valor. Sin embargo, la edición del valor visualmente representaba cierta complejidad, ya que debía depender del tipo de dato de la columna (o campo) seleccionada por el usuario.

Al crear un formulario, el usuario decide los campos que van dentro del mismo, por ejemplo, como se muestra en la figura 9, el campo “Descripción” es de tipo “texto”, y el campo “Participantes” es de tipo “número”, entonces, acorde a la idea principal de la aplicación, que es tener una fuente de datos consistente, el usuario debía ingresar un valor correspondiente al tipo de dato, y si es válido, podrá ser guardado, sin embargo, la aplicación brinda una ayuda para que este objetivo se cumpla mostrando una entrada de datos en relación al tipo escogido por el usuario, cuando este de click sobre el valor de la celda, de la siguiente forma:

- Tipo texto o número: Se muestra una entrada normal de HTML con el parámetro “type” recibido, bien sea “number” o “text”. De esta forma se podía asegurar inicialmente que el valor dentro de la caja de texto iba a corresponder al ingresado por el usuario.

- Tipo fecha: Se muestra una entrada de HTML, enviando en el parámetro “type”, el valor “date”. Gracias a Bootstrap se despliega un calendario, en donde el usuario ingresará la fecha correspondiente.

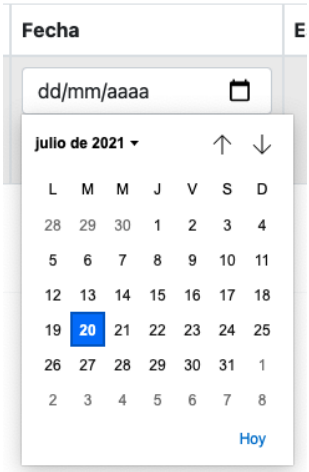


Figura 12: Entrada de formulario tipo fecha. Elaboración propia.

- Tipo lista desplegable: Al crear o actualizar un formulario, el usuario ingresa las opciones que desea desplegar, de tal forma que cuando se quiera editar desde la celda, se mostrará una entrada tipo “select” con las opciones escogidas por el usuario.

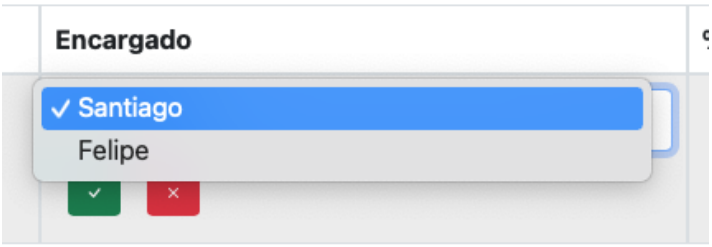


Figura 13: Entrada de formulario tipo selección. Elaboración propia.

- Tipo lista de textos: Se desplegará un diálogo modal que permitirá editar los datos de una lista, que son almacenados en forma de vector de *strings.* Este tipo de datos fue sugerido por el usuario para campos como “Estados” en donde se busca diferenciar visualmente cada entrada de texto, que puede ser ingresada por varios usuarios. Por cuestión de usabilidad, se decidió editar estas celdas utilizando un diálogo modal, para permitir al usuario ver fácilmente todos los datos ingresados anteriormente.

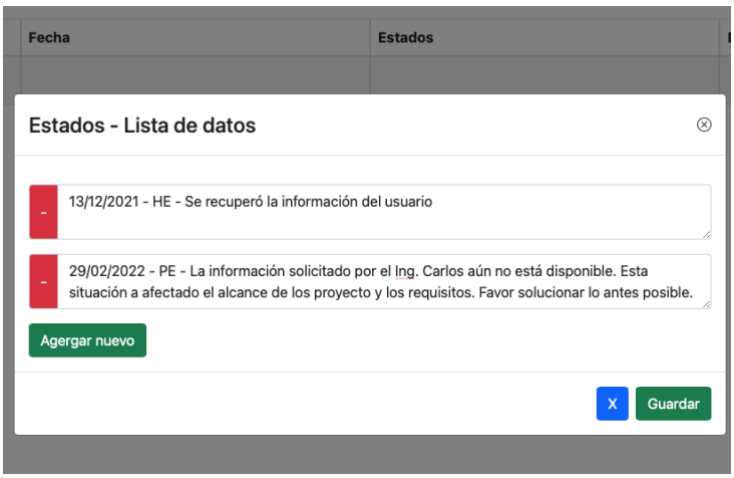


Figura 14: Diálogo modal expuesto para el tipo de campo "lista de datos". Elaboración propia.

Aplicando estos tipos de datos a una celda, se restringe la libertad que tiene un usuario para editar la información en una hoja de cálculo de Google Sheet, que aunque se puede personalizar para validar información o poner un campo "select" de ciertas opciones, algunas personas ingresan datos erróneos o campos vacíos, dejando información inconsistente. Con el uso de la aplicación, al editar información se controla mediante la opción "Requerido" si un campo puede estar vacío, solucionando tal requerimiento.

Para finalizar el presente sprint, se realizó la vista de "listar formularios", en donde se encontraban todos los formularios creados por el usuario. Desde esta ventana se puede seleccionar algún formulario para pasar a la vista de "editar/visualizar información" y visualizar los datos ingresados. En los siguientes sprint, se planeó terminar esta vista para que el usuario visualice la información de determinado formulario, y realice acciones tales como eliminar, clonar y editar.

4.2.3 Sprint 3

El objetivo principal del sprint 3 fue realizar toda la lógica correspondiente a los servicios de GAPI, que consiste en importar y exportar toda la información de una hoja de cálculo de Google Sheet. El conjunto de estas funcionalidades se estimó en 30 horas de trabajo, previniendo posibles retrasos que se podían presentar, ya que no se tenía experiencia en este tipo de servicios que proporciona Google. Por medio de esta funcionalidad el usuario puede importar toda la información de una tabla previamente creada bajo ciertas condiciones, y exportar toda la información, en caso de que se requiera una sincronización de los datos, o se hayan ingresado algunos nuevos, con el fin de que sean mostrados en Google Data Studio.

GAPI es un conjunto de herramientas que pueden ser utilizadas por medio de un REST-Api, CDN (Content Delivery Network), entre otras. Para este caso, fue explorada inicialmente el API que se proveía, pero finalmente representó mayor

complejidad por el acceso OAuth2 que era necesario, por lo se optó por CDN, que consiste en incluir un script de JavaScript que es insertado con la etiqueta <script>, generalmente en el encabezado o en el pie de la página web, creando variables globales que pueden ser accedidas por el código JavaScript cuando se haya ejecutado correctamente por el servidor más cercano al usuario [33]. A través de GAPI es posible acceder a diversas herramientas de la suite de Google, como Google Drive, Google Docs, Google Sheet, entre otras, por medio de una cuenta de Google. Gracias a la elección de utilizar CDN, las funciones cargadas se responsabilizan de la autorización OAuth, desplegando una pantalla con las cuentas de Google del usuario, solicitando permisos para acceder a su información. Cuando el usuario acepta, es posible acceder, por ejemplo, a una hoja de cálculo de Google Sheet a la cual el usuario tiene accesos.

Como recomendación, se explicó al cliente que los documentos de Google Sheet que se iban a utilizar como fuente de datos para Google Data Studio, estuvieran compartidos con todos los usuarios que iban a utilizar la aplicación, ya que si un usuario intenta acceder a una hoja de cálculo a la cual no tiene acceso, la aplicación mostrará un mensaje de error explicando el problema y no se podrá visualizar ninguna información por temas de seguridad.

De acuerdo con la arquitectura de la aplicación, se realizó un componente Redux llamado “spreadsheet” donde se encuentran las acciones que interactúan con GAPI, permitiendo establecer globalmente por la aplicación nuevos estados, en respuesta a las funciones del script CDN. Un ejemplo de los estados generados son los siguientes: se realizó la carga exitosa el script (relacionado con el acceso a internet), la cuenta seleccionada tiene acceso a ciertos documentos de Google Sheet. Desde React se añadió el script en una acción, esto quiere decir que no se ubico como generalmente se utilizaría (hijo de la etiqueta <head>), ya que se necesitaba conocer con exactitud cuándo se había cargado el script para ejecutar un evento en ese momento, y dado que a la función *dispatch* solo se tiene acceso dentro del código de la aplicación, fue necesario realizar esta adaptación.

La primera acción que se realizó fue importar los datos, a través de la función GAPI: `gapi.client.sheet.spreadsheets.values.batchGet()`

Esta función recibe un objeto JSON con ciertos parámetros especificando cómo se iba a obtener la respuesta de la hoja de cálculo, como por ejemplo, por filas, columnas, y entre otras muchas opciones de configuración acorde a las necesidades, y cuando esta era obtenida satisfactoriamente, se procedía a ejecutar la lógica para guardar la información en la base de datos, reconociendo si las filas tenía un ID y asignarla al documento, para finalmente realizar una transacción en Firestore, que es utilizada cuando se requiere actualizar, crear o eliminar un volumen de datos grande sobre una colección, en este caso “data” de un formulario accedido (form). Cuando la acción se había ejecutado, la aplicación actualizaba la información y mostraba los nuevos

valores organizados por su ID junto con una notificación indicando que la tarea se había realizado correctamente.

La segunda acción que se realizó fue exportar datos, a través de la función: `gapi.client.sheet.spreadsheets.values.batchUpdate()`

Esta función recibe un objeto JSON especificando cierta configuración como el rango en el que será escrita la información o si se sobrescribirán las fórmulas que se encuentren, entre otras opciones, y adjunto a ello, los datos que se ubicarán en la hoja. A la información se les aplica cierta lógica para organizarla adecuadamente, de acuerdo al ID asignado, pretendiendo que la hoja de cálculo quede organizada. Cuando la acción finaliza, la aplicación muestra una notificación indicando que la tarea se realizó correctamente.

Cabe resaltar que las respuestas que se reciben a través de las funciones cargadas en el script son considerablemente rápidas, por más información que se requiera escribir o leer en el documento, favoreciendo la interacción con la aplicación. Por otro lado, el componente Redux también contenía otras acciones encargadas de la sesión del usuario que el script controlaba a través de cookies, asimismo funciones para cerrar sesión y cargar inicialmente el script.

Una vez finalizadas las funcionalidades relacionadas con GAPI, se desarrollaron otras historias de usuario que se habían trabajado en sprint anteriores, añadiendo algunos detalles adicionales, uno de ellos relacionado con la usabilidad. Dado que la tabla de datos suele contener más de 12 columnas, era necesario realizar *scroll* hacia un lado, y aunque en el anterior sprint se había trabajado en la posibilidad de ocultar o visualizar algunas columnas, era necesario en ocasiones ver toda la información de la fila, que es desplegada en la pantalla. Por esta razón, se realizó un menú desplegable sobre la primera columna de los datos, de esta forma:

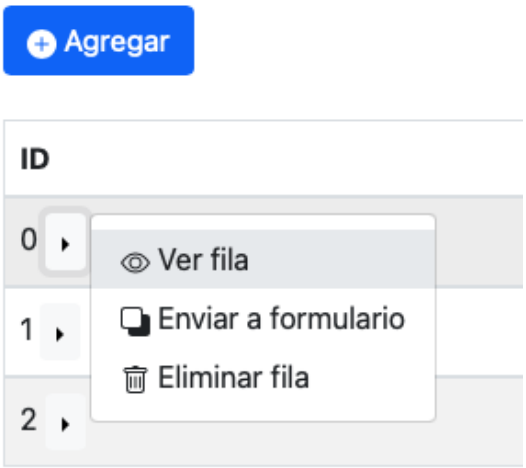


Figura 15: Menú sobre primera columna de la tabla en la vista "editar/visualizar información" - Ítem 1. Elaboración propia.

En el menú, la primera opción “ver fila”, abre un diálogo modal donde se encuentra toda la información de la columna de datos, describiendo campo por campo el valor correspondiente.

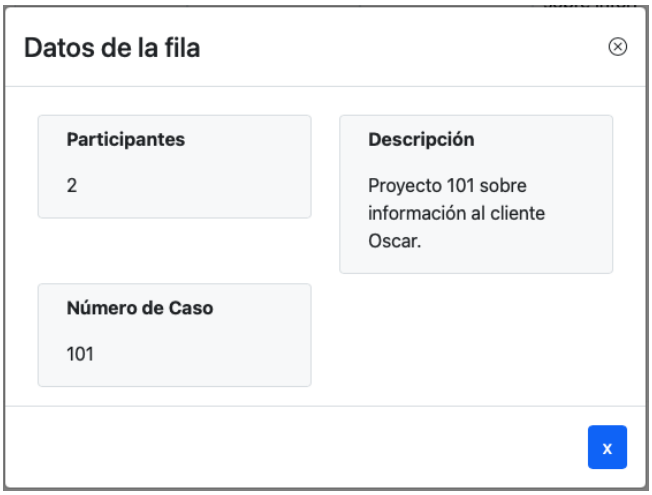


Figura 16: Diálogo modal que expone todos los datos de una fila. Elaboración propia.

Como funcionalidad final del sprint 3, apoyado por Bootstrap se organizó la vista de “listar formularios” creados, obteniendo un resultado visualmente más agradable. Sumado de ello, según las historias de usuario se debía apartar un botón donde el usuario fuera trasladado a la vista de “crear/editar formulario”, de tal forma que se obtuvo lo siguiente:

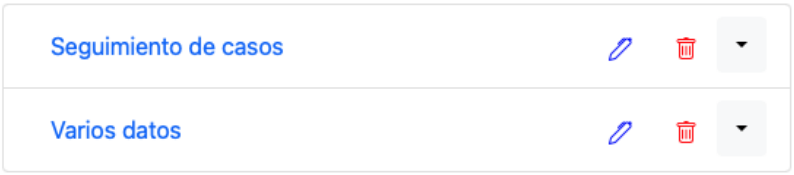


Figura 17: Vista “listar formularios”. Elaboración propia.

Se listan los formularios y el usuario puede presionar sobre su nombre para acceder a la información, o pulsar el botón que contiene un lápiz para editar los campos, sus tipos, el nombre y demás, del formulario seleccionado. La vista de crear y editar es exactamente la misma, lo cual permite que el cliente se familiarice rápidamente con el sistema y sus funcionalidades.

4.2.4 Sprint 4

En el sprint número 4 se realizó una entrega muy importante para el proyecto, ya que se trabajó en el despliegue de la aplicación por medio del hosting gratuito que ofrece Firebase. También se realizaron funcionalidades que hacían lucir a la aplicación en su fase final, como la pantalla del inicio de sesión y la edición de un formulario, que hasta el sprint anterior el usuario era trasladado a la ruta desplegando el componente, pero la lógica fue establecida en el presente sprint.

Inicialmente, se trabajó sobre la vista de “crear/editar formulario”, donde los datos ya se encontraban cargados por el manejo de Formik, y lo requerido consistía en enviar la información de todos los campos a actualizar, como su nombre, Google Sheet ID y campos con sus tipos. Cuando el usuario termina la actualización, presiona el botón de actualizar y se ejecuta un evento que llama una acción la cual no modifica el estado de la aplicación, pero si realiza un llamado a la base de datos, de tal forma que cuando se recibiera una respuesta exitosa, condujera al usuario a la vista de “listar formularios”, donde si lo requería, podía visualizar su nuevo nombre, o campos incluidos o editados.

Por otro lado, en la pantalla “editar/visualizar información”, con el objetivo de mejorar la usabilidad, se trabajó en columnas reajustables, las cuales permiten modificar su ancho. Esto entrega al usuario una mejor experiencia con la aplicación ya que pueden existir campos que contengan muy poca información en la celda, como un único número, o campos de texto muy amplios, donde se almacene, por ejemplo, el contenido de un correo electrónico. Aunque la librería react-table ofrecía cierta ayuda para incluir las columnas reajustables, se ocuparon considerables horas de trabajo para cubrir este requerimiento.

En la misma vista mencionada anteriormente, se trabajó en la opción “eliminar fila” del menú expuesto en la figura 15. Similar a otras funcionalidades, se ejecuta un evento que llama una acción para enviar una petición a la base de datos donde se elimina el documento, perteneciente a la colección “data” del formulario seleccionado. Estos mismos pasos se utilizaron para la opción de “eliminar formulario”, que es mostrada al usuario a través de un icono similar a un bote de basura, visto en la figura 17. Desde este botón se enviaba una petición a la base de datos para eliminar el documento perteneciente a la colección “forms”.

Para poder entregar una aplicación con un flujo similar al que va a realizar el usuario, en este sprint se realizó toda la gestión que permitía iniciar sesión a través de un único método, el cual era google sign-in. Inicialmente se trabajó en una interfaz con una apariencia amigable para el usuario y agradable a la vista, que contenía los logos de las organizaciones y un único botón.

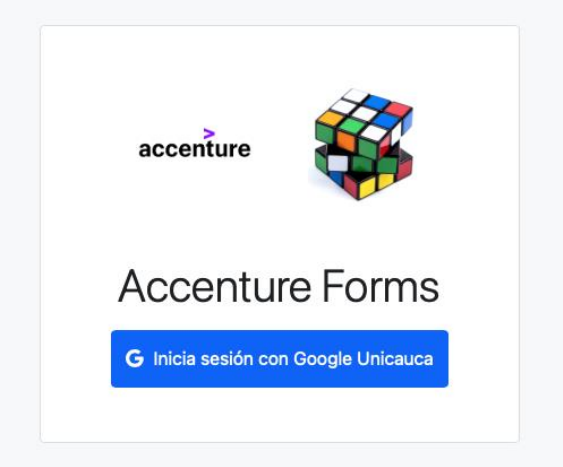


Figura 18: Vista "login". Elaboración propia.

Se construyó el único componente Redux faltante que gestionaba la sesión del usuario, llamado Auth, en el cual se controlaban tres acciones: Google login, Iniciar sesión y cerrar sesión.

- A través de la acción de Google Login, que era ejecutada por el botón visible en la imagen anterior, se despliega una ventana auxiliar en la cual el usuario ingresa su cuenta de Google con sus respectivos datos, iniciando sesión en la aplicación, sin embargo, se controlaba el dominio del usuario que accede, que debe ser estrictamente "cubo.com", ya que si no era así, la aplicación no permite continuar, mostrando un error al iniciar sesión, pero si es permitido, se ejecuta la acción de iniciar sesión.

- A través de la acción Iniciar sesión se controla el estado de la aplicación, almacenando si existe la sesión del usuario. Esta acción es ejecutada en dos ocasiones: cuando se inicia sesión por google y el usuario se ha validado permitiendo su acceso, y cuando un usuario que ya ha iniciado sesión, ingresa nuevamente a la aplicación web, el cual puede ser redirigido a una vista protegida gracias al *router* de la aplicación y a una sesión válida, evitando que el usuario tenga que iniciar sesión cada vez que ingresa al aplicativo.

- A través de la acción Cerrar sesión, se utiliza una función proveída por la librería de Firebase para borrar el registro de la cookie y cambiar el estado de la aplicación borrando la sesión del usuario

Con la creación y finalización del componente de autorización, y la vista que permite al usuario iniciar sesión, se finalizan todas las pantallas de navegación, permitiendo realizar la entrega de avances con un producto muy similar al resultado final. Así es como para terminar el sprint 4 se trabajó en el primer despliegue público de la aplicación permitiendo que el usuario pueda acceder desde su navegador a la página web. Firebase brinda herramientas como Firebase CLI para realizar estos despliegues públicos fácilmente, ya que cada vez que se programaba una entrega en los

siguientes sprint, se realizaba un despliegue para que el usuario interactuara con el producto.

4.2.5 Sprint 5

El sprint 5 se enfocó en aspectos importantes de la usabilidad de la aplicación. En la reunión de seguimiento del sprint 4, se acordaron algunas modificaciones a los requisitos del sistema ya que fue posible que el usuario interactuara con la aplicación. Las modificaciones se ajustaban al alcance, y estaban relacionadas con la usabilidad, junto con algunos requerimientos nuevos que complementaban la aplicación. También se trabajó en actividades planeadas en el documento inicial.

En la vista de “editar/visualizar información”, debido a la cantidad de columnas y filas que puede tener una tabla, el volumen de datos hace que se consuman muchos recursos y el rendimiento de un equipo de cómputo se vea afectado, por esta razón, se acordó con el cliente un nuevo requerimiento que consiste en aplicar una paginación flexible, que permitiera al usuario seleccionar un rango de valores.

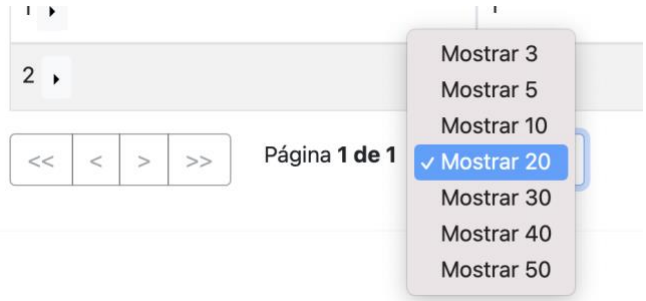


Figura 19: Sección de paginación en la vista “editar/visualizar información”. Elaboración propia.

La imagen anterior muestra el resultado del requerimiento, en donde se incluyeron botones que permiten navegar dentro de las páginas y un campo tipo “select” para elegir la cantidad de datos de cada página. Sumado a ello, se incluyeron “query params”, que consiste en agregar variables sobre una ruta, para este caso, la ruta actual con el objetivo de optimizar la paginación y permitir al cliente, si lo desea, guardar como un marcado dentro del navegador, la URL incluyendo la página y el tamaño, aportando a la usabilidad. Cabe resaltar que la librería react-table brinda ciertas funciones para utilizar paginación.

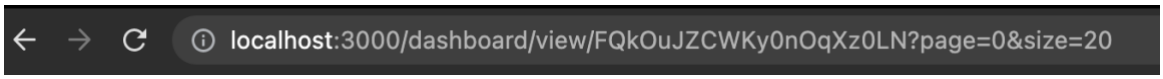


Figura 20: URL de ejemplo en la vista con la inclusión de query params. Elaboración propia

Otro nuevo requerimiento hablado con el cliente incluye la funcionalidad de filtrar valores dentro de una o varias columnas, por ello, se creó un botón que despliega una entrada de texto en cada columna para que el usuario pueda ingresar un valor, y

filtrar dentro de la información de la tabla. Se acordó crear una entrada de texto que permitiera ingresar cualquier número o cadena, dando como resultado valores que contengan la información digitada. Era posible configurar librería para filtrar según: contenían el valor ingresado, iniciaba con el valor, finalizaba con el valor, no contenía el valor, sin embargo, para no aportar complejidad en el requerimiento, se definió la configuración de “contener el valor” por defecto para todas las columnas.

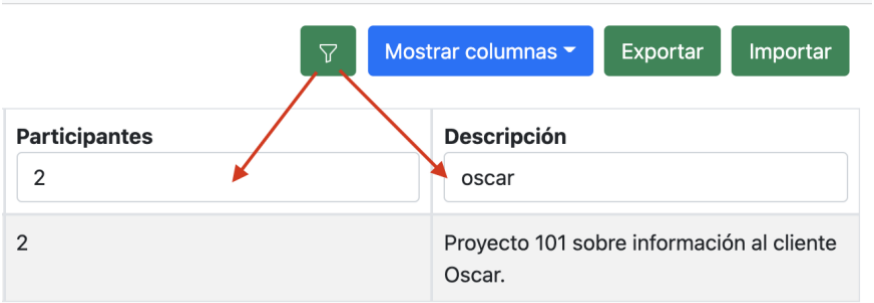


Figura 21: Despliegue de entradas de texto en cada columna. Elaboración propia.

Dado que, al finalizar el presente sprint, la aplicación ya debía estar terminada según la estimación inicial, existían algunas actividades previamente definidas, sin embargo, en consecuencia a la inclusión de nuevos requisitos, se incrementó la fecha de entrega del proyecto dos semanas más que fueron dialogadas con el cliente, el cual no tuvo problema en aceptar, ya que constantemente observaba los avances y era notable la aceptación del producto. Una funcionalidad que se encontraba prevista para finalizar la aplicación y ultimar detalles, es la inclusión de retroalimentaciones visuales que muestran al cliente que algún proceso se está realizando y debe esperar a su terminación, generalmente utilizando componentes como *spinners,* los cuales fueron incluidos en cada botón que ejecutaba una acción, indicando que el usuario debía esperar cierto tiempo. Esta retroalimentación se utilizó para eventos como: exportar e importar información a Google Sheet (lo cual puede tardar más de 6 segundos en ejecutarse), agregar o borrar una fila de datos, crear o actualizar un formulario. Estas acciones involucran la base de datos y el tiempo de espera es variable.

Figura 22: Ejemplo de retroalimentación al usuario utilizando spinners en un diálogo modal. Elaboración propia

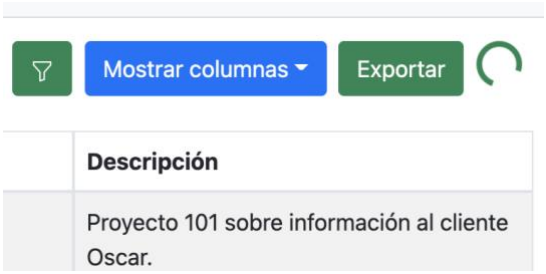


Figura 23: Ejemplo de retroalimentación al usuario utilizando spinners en la vista "editar/visualizar información". Elaboración propia

Finalmente, relacionado con la usabilidad y las actividades planeadas inicialmente, se incluyó un botón que permitía mostrar u ocultar toda la información de una celda en caso de que tuviera demasiada información:

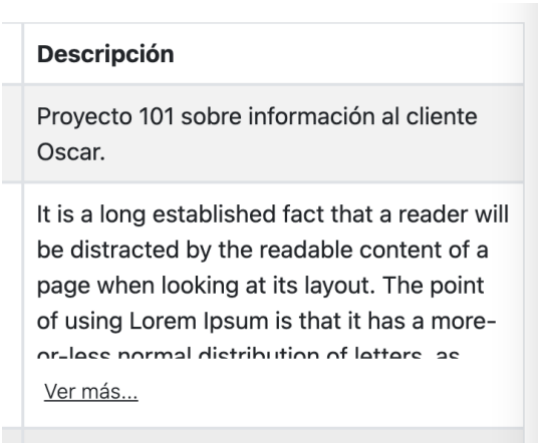


Figura 24: Botón "Ver más…" sobre una celda. Elaboración propia

Como se puede observar en la imagen, la segunda celda tiene más información que la primera. Sin el botón de “ver más” las celdas tendrían un tamaño muy diferente desplegando toda la información, haciendo que la visualización se dificulte, sin embargo, creando el botón, todas las celdas tienen el mismo tamaño al cargarse la página por primera vez, y si se desea, se puede filtrar una celda en especifico, aumentar el ancho y visualizar toda la información. En la parte inferior de la celda, se muestra el botón de “ver menos” para cuando se haya leído la información.

El cliente solicitó dos nuevos requisitos para complementar la aplicación los cuales consisten en: clonar un formulario y mover o copiar una fila de un formulario a otro. Para realizar la acción de clonar un formulario, se creó un menú desplegable a un lado de las acciones editar y eliminar cuando se listan los formularios creados. La clonación se hacía respecto al nombre, el Google Sheet ID, los campos y sus tipos, pero no de la información, ya que como lo mencionaba el cliente, esta funcionalidad es útil cuando dos formularios tienen campos similares o iguales, pero la información dentro de ella es diferente.

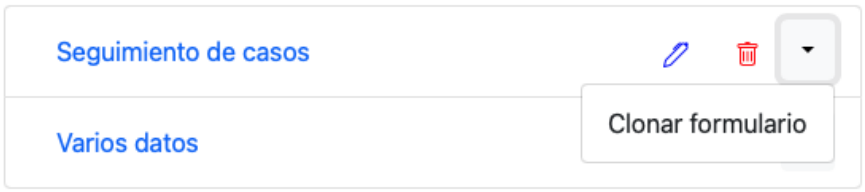


Figura 25: Despliegue del menú de un elemento en la vista “listar formularios”. Elaboración propia

Sobre la tabla de datos, en la primera columna, donde se había creado ya un menú desplegable (ver figura 26), se incluyó un ítem para enviar la fila de datos a otro formulario, en relación con el siguiente requisito, con la opción de mover o copiar. Para esto, fue necesario crear un diálogo modal donde el usuario seleccionará el formulario de destino.

Figura 26: Menú sobre primera columna de la tabla en la vista “editar/visualizar información” - Ítem 2. Elaboración propia.

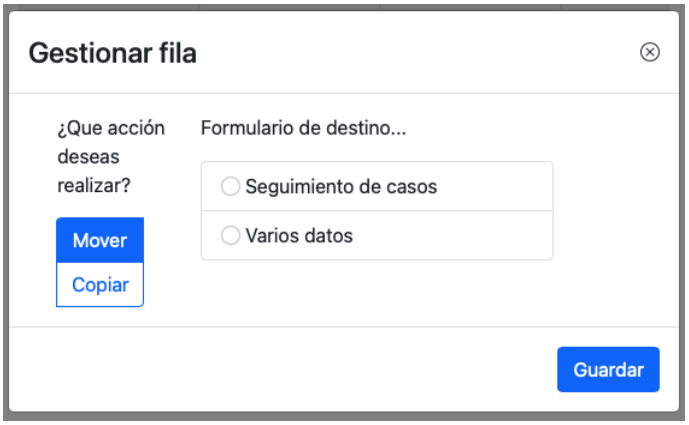


Figura 27: Diálogo modal para mover o copiar una fila de datos. Elaboración propia.

Después de presionar el botón “guardar”, se envía la fila al formulario seleccionado, añadiendo a la información que tiene en el momento. Del usuario depende enviar los datos a un formulario con columnas similares, sin embargo, no es relevante el tipo del campo, ya que la aplicación es lo suficientemente flexible para mostrar o editar información de cualquier tipo en la celda. Gracias a esta nueva funcionalidad, el usuario puede tener un respaldo de los datos, o gestionar su información por relevancia, roles y demás, en diferentes formularios, por ejemplo, el usuario puede tener dos formularios con las mismas columnas gracias a la clonación, y mover la información de un lugar a otro, de acuerdo con un responsable en específico, una fecha o un campo a su conveniencia.

Un requisito realizado en el primer sprint relacionado con ocultar o visualizar una columna fue reajustado, ya que se trabajó en la actualización de un formulario. Con fines de usabilidad, las columnas que el usuario ocultaba y mostraba eran guardadas localmente a través del *local storage*, para cuando el usuario reingresara a la aplicación, la personalización de la tabla no se perdiera, sin embargo, cuando un formulario se actualizaba, era posible cambiar de nombre a una columna (campo), eliminarla o cambiar su tipo. Por esta razón, se trabajó en una sincronización entre lo almacenado en la base de datos y los datos guardados localmente en el navegador del cliente cada vez que se accedía a un formulario.

Para presentar al cliente la aplicación con datos reales, se realizó una copia de unos de los documentos que el usuario frecuentemente utiliza, y se creó un formulario en la herramienta para importar los datos de Google Sheet. Los resultados fueron de agrado para el cliente, ya que visualizó los cambios realizados con información real.

4.2.6 Sprint 6

En el sprint 6 se realizaron dos actividades relacionadas con el aseguramiento de calidad, ya que el producto estaba pronto a entregarse. Dado que no existía un equipo

de pruebas para este proyecto, se designaron algunas horas de trabajo en compañía de un potencial usuario para realizar pruebas a nivel global de la aplicación.

Una de las funcionalidades relacionadas a la finalización del proyecto consiste en aportar una retroalimentación al cliente si una tarea se realizó o no. Un error muy común dentro de la aplicación es acceder al inicio de sesión mediante la ventana de GAPI, con una cuenta de google que no tiene permisos de escritura o lectura sobre un archivo. Por esta razón, la aplicación despliega notificaciones de éxito o error dependiendo de la acción, de la siguiente manera:

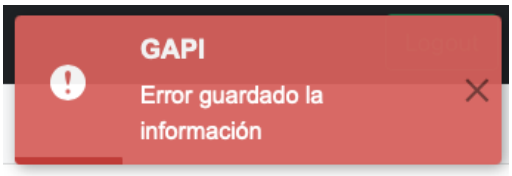


Figura 28: Notificación de error. Elaboración propia.

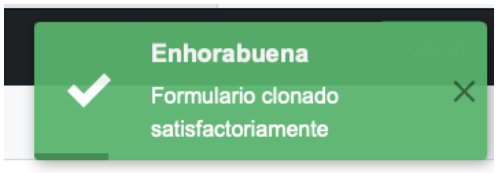


Figura 29: Notificación de éxito. Elaboración propia.

Estas notificaciones se muestran generalmente en acciones en las que el usuario tiene que esperar cierto tiempo para saber el resultado de la operación. Los errores son explicados en el manual de la aplicación.

En la base de datos de Firebase, exactamente Firestore es posible configurar reglas de uso, que permiten crear, eliminar o modificar información de un documento sobre una colección, de acuerdo con una sesión activa, una fecha, una colección en específico, entre otras. Para este caso, existía una colección llamada “forms”, y dentro ella, cada documento contenía datos y otra colección llamada “data”. La única regla necesaria que se configuró por razones de seguridad consistió en permitir cualquier acción sobre “forms” siempre y cuando el usuario envíe una petición que tenga una sesión activa. La sesión es enviada en la cabecera de cada petición, de lo cual se encarga la biblioteca de Firebase, y desde su página web, se administran las reglas activas.

En la vista de “crear/editar formulario”, fue necesario agregar dos campos que se deben almacenar en la base de datos, los cuales eran “Hoja” y “Contador ID”. Estos nuevos campos se incluyeron luego de una conversación con el cliente en la cual mencionó que se estaba tomando por defecto la primera hoja del documento de Google Sheet, por esta razón se incluyó el campo “Hoja”, para que el usuario indique sobre cual hoja del documento desea importar y exportar la información. Por otra parte, cuando el usuario importaba la información, el identificador de las filas no se

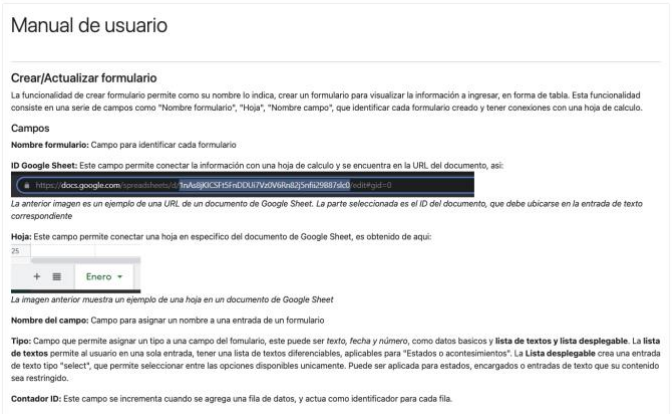
incrementaba, por lo tanto, al ingresar una nueva, el ID se podía repetir. Para solucionar este problema, se incluyó el campo “Contador ID”, el cual representa el identificador que contendrá la siguiente fila a ingresar, de tal forma que, si el usuario importa cierta cantidad de datos, puede editar el valor para determinar cuál será el próximo identificador de la fila a ingresar.

Finalmente, se realizaron algunas pruebas del flujo principal que tendría un usuario al utilizar la aplicación. Fue detectado un error básico al importar información de un documento, sin embargo, fue corregido en el momento. Para evitar el sesgo que se podría tener desde el rol de desarrollador y probar la aplicación, se tuvo el apoyo de Felipe Pieschacon, mencionado anteriormente como interesado del proyecto, que utilizaría el sistema cuando estuviese en producción.

4.2.7 Sprint 7

En el último sprint de trabajo se realizó un requerimiento muy importante que consistía en elaborar el manual del usuario que correspondía a una condición de entrega. En la última semana se realizaron capacitaciones con usuarios de la aplicación y se modificó el archivo Markdown con el que cuentan generalmente todos los proyectos de desarrollo.

En conversaciones con el cliente, se acordó que el manual de usuario estaría ubicado dentro de la aplicación, lo cual facilita al usuario acceder fácilmente a través de la barra de navegación si surge alguna duda sobre una funcionalidad, sumado a que se trató de elaborar un manual lo mas detallado posible, sobre todas las acciones que se pueden realizar.


Manual de usuario

Crear/Actualizar formulario

La funcionalidad de crear formulario permite como su nombre lo indica, crear un formulario para visualizar la información a ingresar, en forma de tabla. Esta funcionalidad consiste en una serie de campos como "Nombre formulario", "Hoja", "Nombre campo", que identificar cada formulario creado y tener conexiones con una hoja de calculo.

Campos

Nombre formulario: Campo para identificar cada formulario

ID Google Sheet: Este campo permite conectar la información con una hoja de calculo y se encuentra en la URL del documento, así:

https://docs.google.com/spreadsheets/d/1nAsljKlCSFt5FnDDUi7Vs0V6lRnB2j5nfii29887slc0/edit#gid=0

La anterior imagen es un ejemplo de una URL de un documento de Google Sheet. La parte seleccionada es el ID del documento, que debe ubicarse en la entrada de texto correspondiente

Hoja: Este campo permite conectar una hoja en especifico del documento de Google Sheet, es obtenido de aqui:

25 Enero

La imagen anterior muestra un ejemplo de una hoja en un documento de Google Sheet

Nombre del campo: Campo para asignar un nombre a una entrada de un formulario

Tipo: Campo que permite asignar un tipo a una campo del fomulario, este puede ser *texto*, *fecha* y *número*, como datos basicos y **lista de textos y lista desplegable**. La **lista de textos** permite al usuario en una sola entrada, tener una lista de textos diferenciables, aplicables para "Estados o acontesimientos". La **Lista desplegable** crea una entrada de texto tipo "select", que permite seleccionar entre las opciones disponibles unicamente. Puede ser aplicada para estados, encargados o entradas de texto que su contenido sea restringido.

Contador ID: Este campo se incrementa cuando se agrega una fila de datos, y actua como identificador para cada fila.

Figura 30: Vista previa del manual de usuario. Elaboración propia

Fueron realizadas capacitaciones a usuarios que utilizarían la aplicación, entre ellos, los nuevos practicantes que iniciaron para el segundo semestre del 2021. Por medio de videollamadas y utilizando la última versión de la aplicación, fueron gestionadas e impartidas las capacitaciones con base al manual de usuario. En equipo, se realizó el paso a producción de la aplicación, creando un respaldo de todos los documentos de

Google Sheet, para que la aplicación, a partir de ese momento, gestionara los documentos, y como primer paso, se realizó la importación de toda la información de estas hojas de cálculo. Para ese momento, ya el equipo empezaba a modificar el estado de las incidencias (filas) diariamente.

Finalmente, se realizó una modificación al archivo Markdown (.md) donde se describió el proyecto, cómo ejecutarlo, y qué tecnologías se utilizaron para su desarrollo, considerando que el proyecto es escalable y en caso de que se quiera modificar o aumentar alguna funcionalidad, resulte fácil para un desarrollador React orientarse en el marco del proyecto.

4.2.8 Manejo de Git

El proyecto fue almacenado a lo largo de su desarrollo en una de las plataformas más utilizadas en la actualidad para el control de versiones, organizado de la siguiente manera:

- Todo *commit* o rama, debía contener obligatoriamente el identificador de la tarea o historia de usuario otorgado por Jira al agregarla en el *backlog*.
- Existieron dos ramas dentro del proyecto, “master” y “develop”. Sobre la rama *develop*, se encontraban todos los cambios realizados entre semana. Sobre la rama master se realizaba un *Pull Request* al final de cada sprint con la aplicación completamente estable.
- En cada *commit*, se debía explicar brevemente cuáles fueron los cambios realizados.

Este control sobre la plataforma permitía tener el proyecto organizado, ya que, si alguna funcionalidad dañaba el sistema, se tendría un respaldo detallado igualmente en caso de que otro desarrollador trabajara en el proyecto.

Capítulo 5

5. Análisis de resultados

5.1 Banco Icol

En el banco Icol fue posible hacer parte de un equipo de desarrollo, dentro de un proyecto de transformación digital donde se aplicaban las metodologías ágiles y TDD para aportar a la creación de un software bancario. Todas las funcionalidades asignadas fueron desarrolladas exitosamente con el monitoreo de un desarrollador con varios años de experiencia, con el fin de reconocer un proyecto de desarrollo a gran escala y aplicar diferentes técnicas o procesos que se llevaban a cabo en el desarrollo, con un cliente que presta sus servicios en Colombia, pero tiene operación a nivel mundial, solicitando software bancarios de gran tamaño y complejidad, en los cuales interactúan múltiples profesionales. Una vez entendidas la mayoría de las responsabilidades, se inició el apoyo en los requerimientos solicitados.

Junto con esta persona se realizó el acompañamiento para aplicar adecuadamente el desarrollo guiado por pruebas conforme a la ejecución de los requerimientos, y se aplicaron mecanismos para mejorar la calidad del software mediante la refactorización propuesta por TDD. También se realizó la documentación de múltiples secciones del código fuente, la cual resultaba ser muy concisa, debido a las indicaciones del líder técnico, describiendo únicamente lo esencial con las ayudas que brinda el entorno de desarrollo integrado (IDE) sobre los lenguajes de programación Typescript y JavaScript, utilizados ampliamente en el desarrollo de la práctica profesional.

Cada historia de usuario, tarea o requerimiento realizado representó un avance sobre el producto final, el cual es el resultado del trabajo de un equipo que tiene como responsabilidad el proceso de transformación digital que desea realizar el cliente sobre sus procesos internos y servicios hacia sus usuarios, apoyado por Accenture, creando para este caso, una aplicación que evite al usuario del banco realizar ciertas transacciones presenciales.

Pertenecer al equipo implicó poner en práctica las metodologías ágiles, bajo el marco de trabajo Scrum, desde el rol de integrante de un equipo de desarrollo, en donde se cubrieron todas las responsabilidades, como asistir y aportar a las reuniones diarias de seguimiento, informar progreso, trabajar en equipo, además de reportar frecuentemente las actividades realizadas en la herramienta Jira, para que el grupo de gestión de proyectos presentará los avances, acorde a lo dialogado con el cliente.

Desde esa perspectiva fue posible observar cómo la comunicación con el cliente representaba un problema tanto para la metodología, como para el proyecto, ya que lograr dialogar con las personas encargadas del cliente resultaba difícil, teniendo en

ocasiones que obtener una cita a través de un portal interno del banco, que podía tardar de 1 a 8 días hábiles, en caso de que fueran situaciones que no se trataran dentro de los seguimientos semanales. Sin embargo, por parte del Accenture, es destacable la comunicación dentro del grupo de desarrollo que permitía poner en práctica las actividades propuestas por el marco de trabajo.

Pese a los problemas de comunicación, fue posible entregar una aplicación con altos estándares de calidad, realizada por un grupo de trabajo que desarrolla, prueba y documenta cada requisito.

La tabla 2 presenta una visión general de la necesidad del cliente por la cual se ejecuta el proyecto donde se participó:

Necesidad del cliente	La necesidad identificada que busca atender el proyecto (desde la perspectiva del rol ejercido), consistió en realizar una transformación digital de procesos externos de cara al cliente para que los usuarios tengan la capacidad de realizar transacciones desde una aplicación móvil y no dirigirse a las instalaciones físicas.
Intervención de la empresa	Desarrollo y mantenimiento de la aplicación móvil
Situación esperada después de la intervención	Los usuarios realizan múltiples transacciones bancarias a través de la aplicación, evitando aglomeraciones en las instalaciones físicas.

Tabla 2: Necesidad del cliente e intervención

La tabla 3 presenta un esquema resumido de la principal actividad realizada con el banco Icol:

Situación anterior	Requisitos por implementar en el área de Frontend, relacionados con la interfaz de usuario, testing y la invocación a las funcionalidades del Backend.
Intervención	Apoyo en la solución de múltiples historias de usuario.
Situación después de la participación	Requisitos implementados que integran en el producto final y que son presentados al cliente al finalizar cada sprint.

Tabla 3: Actividad principal - Banco Icol

La tabla 4 expone la principal dificultad presentada en la participación del proyecto:

Dificultades presentadas	Estrategias de solución
Baja comunicación y participación del cliente	Desafortunadamente el cliente no estaba interesado en tratar de resolver el problema, mientras que el equipo de Accenture aportaba más de lo necesario para la aplicación de las metodologías ágiles, sin embargo, se realizó una recomendación teniendo en cuenta el rol ejercido, para que se implementara una nueva estrategia de comunicación que tenga en cuenta los componentes recomendados por el marco de trabajo Scrum

Tabla 4: Dificultad presentada - Banco Icol

5.2 Banco Cubo

En el banco cubo se realizó el apoyo en diferentes actividades de gestión de proyectos. La mayoría de ellas, estaban relacionadas con la presentación de evidencias de los trabajos realizados por los equipos de desarrollo. Estas actividades estaban dirigidas por profesionales con estudios como ingeniería industrial, y en cada ocasión que fuese necesario, se apoyaba en la ejecución de actas de reuniones o seguimientos, monitoreando calendarios, construyendo informes de progreso, entre otras.

Además de realizar estas actividades, la razón principal para apoyar en dicho rol consistía en innovar a través de herramientas tecnológicas algunos procesos de la gestión de proyectos. Por esta razón, y como solución temporal, se realizaron ajustes en las hojas de cálculo de uso frecuente de Google Sheet, tratando que las personas que gestionan la información lo realizaran de manera correcta, a través de los múltiples servicios ofrecidos por la herramienta, haciendo más fácil la edición de los datos para su frecuente actualización.

De esta forma, se podría asegurar una mejor presentación del trabajo realizado en las reuniones de seguimiento que se efectuaban con el cliente, gracias a que los dashboards existentes en el momento, contaban con una fuente de información mejor estructurada, y junto con las capacitaciones impartidas, los ingenieros tenían conocimiento de los datos que debían ser ingresados en cada columna, y podían realizar esta tarea más fácilmente. Algunas hojas de cálculo se exponían al cliente gracias a que eran visualmente entendibles, sin embargo, el cambio constante entre aplicaciones no resultaba ser la mejor forma para exponer avances.

Cuando se inició el trabajo con el banco Cubo, tomó un tiempo considerable reconocer el contenido de varios archivos y su ubicación, los cuales se actualizaban constantemente. Incluso algunas personas que ofrecían las capacitaciones tenían dificultades para exponer con claridad la información y para qué se utilizaba. Como

solución de este problema, se realizó la página web en Google Sites, permitiendo a cualquier miembro del equipo, visualizar con facilidad los enlaces en donde se encontraba cierto dashboard de Google Data Studio, con su respectiva fuente de datos. Esta solución resulta ser escalable, ya que la herramienta permite incluir a la página web diversos componentes, para visualizar directamente los dashboard, realizar una descripción de los enlaces, incluir imágenes, texto, y diversas funciones de forma gratuita.

De acuerdo con el plan de trabajo construido, la siguiente actividad consistía en mejorar la presentación de evidencias de trabajo, a través de la herramienta Google Data Studio. Para esto, se crearon una serie de dashboard los cuales contaban con un patrón de diseño con criterios mínimos de usabilidad, que permiten al usuario orientarse en la herramienta sin importar en cual dashboard se encuentre. Esto fue de gran ayuda permitiendo presentar al usuario un documento mucho más organizado y claro, donde se exponía la información.

Como se indicó anteriormente, la actividad de optimizar las hojas de cálculo se realizó como una solución temporal que fue resuelta con la ejecución y entrega del software Accenture Forms construido en los últimos dos meses de la práctica. Con la ayuda del software se podía asegurar que los requerimientos de editar y crear información, se realice de una forma más estructurada. A través de la vista de edición, cualquier campo de la tabla resultaba fácil de modificar, de acuerdo con el tipo de datos que se había asignado a cierto campo o columna. De igual forma para crear una fila de datos nueva, en la cual se verificaban los tipos de datos ingresados e incluso campos vacíos, de acuerdo con cada formulario personalizado creado por cualquier usuario.

Todo ello permitía exportar la información a una hoja de cálculo de Google Sheet, enviando datos consistentes para que Google Data Studio desplegara esta información sin errores al cliente. Todos los datos eran almacenados en una base de datos, lo cual hace escalable el proyecto, ya que esta información se puede utilizar posteriormente para realizar mejoras.

La creación de la herramienta favorece la confiabilidad en el flujo de la información. La persona ingresa información validada, dos herramientas la almacenan, y una última herramienta consume esta información, y permite visualizarla de forma organizada. A través del uso o creación de todas las herramientas mencionadas anteriormente, se aportó a ciertos procesos pertenecientes a la gestión de proyectos con el banco Cubo, a través de las herramientas que proporciona Google, optimizar tiempo y recursos, que se verán reflejados en un futuro.

Las siguientes tablas presentan un esquema resumido de las actividades realizadas con el banco Cubo:

Situación anterior	Integrantes de los equipos de desarrollo no diligencian correctamente la información de sus proyectos en los documentos de seguimiento en Google Sheet
Intervención	Optimización de las hojas de cálculo con los servicios proporcionados por la herramienta
Desafío presentado	El grupo de trabajo con el banco Cubo era considerablemente grande y se debían impartir apropiadamente las capacitaciones donde se explicaran los nuevo cambios para el efectivo ingreso de información
Situación después de la participación	Ingreso de datos más confiables y validados que facilitan la actualización para su exposición al cliente.

Tabla 5: Esquema resumido - Actividad Google Sheet

Situación anterior	El Application Manager presenta información desactualizada, errónea y desorganizada acerca del seguimiento de los proyectos en las reuniones sostenidas con el cliente.
Intervención	- Creación de plantillas y Dashboards en Google Data Studio para exponer la información de una forma más organizada. - Creación de la herramienta Accenture Forms para establecer una solución definitiva al problema del ingreso de la información de cada proyecto (seguimiento).
Situación después de la participación	El Application Manager menciona que las reuniones son más objetivas y efectivas.

Tabla 6: Esquema resumido - Actividad Google Data Studio y Accenture Forms

Situación anterior	Integrantes de los equipos de desarrollo presentan dificultades para acceder a todo el volumen de información a los archivos más importantes y distinguirlos.
Intervención	Creación de la pagina web a través de Google Sites
Situación después de la participación	Los involucrados ingresan más rápidamente a los documentos de frecuente actualización, y a través del contenido editable, resulta más fácil la actualización de la URL de un documento o poder describir cada uno e ingresar contenido multimedia.

Tabla 7: Esquema resumido - Actividad Google Sites

Capítulo 6

6. Conclusiones

A lo largo del desarrollo de la práctica profesional se identificaron una serie de procesos, los cuales podía requerir una mejora o apoyo, y de esta forma contribuir a la construcción de un software, hacer parte de la transformación interna o externa de los clientes e igualmente apoyar las distintas actividades de gestión de proyectos. Por algunas restricciones de seguridad, clientes con procesos obsoletos, problemas de comunicación y demás dificultades, en ocasiones la labor era más difícil de realizar, sin embargo, se logró aportar valor a los procesos y los productos de los clientes, y observar los beneficios de aplicar ciertas metodologías, practicas y tecnologías en el proceso de construcción.

La practica profesional inicia dando a todos los practicantes una introducción a sus labores y los derechos y deberes de trabajar con Accenture para posteriormente entrar a sus labores diarias. Este proceso de incorporación a un entorno empresarial para un estudiante de la Universidad del Cauca puede llegar a ser complejo en un inicio, ya que se trabaja en proyectos reales de gran magnitud, con equipos multidisciplinarios considerablemente grandes y exigentes, que llevan años en ejecución y mantenimiento. Probablemente uno de los retos más importantes es conocer que el grupo de trabajo está conformado por profesionales de múltiples especialidades que monitorean el progreso de cada practicante, su apropiada o problemática inclusión al proyecto, su rendimiento, comunicación, entre otros temas que no son comunes en la carrera universitaria, ya que, probablemente no existió una relación con estudiantes de otras áreas, desde las futuras responsabilidades profesionales de ambas partes (más allá de las éticas que son igual de importantes).

Cabe aclarar que la complejidad del proceso de incorporación al ámbito laboral se relaciona con las habilidades blandas y técnicas por igual que el estudiante haya adquirido a lo largo de su trayectoria universitaria, ya que la inclusión a otros ambientes no académicos que le ayuden a crecer intelectualmente, sumado a todos los conocimientos adicionales que ha adquirido fuera de la academia por sus propios medios, hace que este proceso complejo sea más fácil de abordar.

Es de resaltar que realizar una practica profesional como modalidad de grado, con una empresa con varios años en el mercado que contrata periódicamente más de 50 practicantes con un determinado plan de trabajo, permite al estudiante obtener una amplia introducción en cuanto a proyectos reales a gran escala, más allá de los posibles clientes locales con los que se trabaja en asignaturas del plan de estudios de Ingeniería de Sistemas como Proyecto I y II, agregando valor a su crecimiento profesional al enfrentarse a nuevas responsabilidades considerablemente exigentes.

Crear esta modalidad de grado permite de igual forma a los estudiantes que se enfrentan a las necesidades actuales de los proyectos entregar una retroalimentación a la universidad, en relación con los conocimientos o experiencias enfrentadas. De esta forma, se considera que una recomendación importante para constatar en el presente documento es que los egresados que están interesados en el área del desarrollo terminan sus estudios sin conocer en profundidad prácticas metodológicas actuales como TDD, que involucran aplicar conceptos relacionados con las pruebas unitarias y de integración que son indispensables en este tipo de proyectos. Si bien es imposible abarcar toda la lógica ya que esta puede depender del lenguaje de programación que se este utilizando, existen algunos principios básicos comunes que no se revisaron en la carrera universitaria, como la inclusión de bibliotecas de *testing* como Jest, donde se utilizan mocks y stubs, también un análisis adecuado que permita determinar si realmente se está probado correctamente determinada función o funcionalidad, entre algunos otros conceptos, como ambientes de desarrollo y pruebas, etc.

De igual forma si el estudiante tiene interés por cierta área dentro de la ingeniería de software, algunos procesos se han renovado permitiendo que no sean tan rígidos, por la inclusión de las metodologías ágiles y lo que implica (como la flexibilidad en los requisitos). incluida toda la cultura de DevOps y Scrum como marco de trabajo. Se considera importante continuar con la inserción de estos conceptos o temáticas que poco a poco se han incluido como electivas en los últimos años.

Por parte de la compañía, gracias a la organización logística de los practicantes, se logra entregar productos valiosos que son coordinados por personas con amplia experiencia e incorporan a profesionales y practicantes que se encuentran en el proceso de crecimiento, y de esta forma, el apoyo de este segundo grupo de personas se convierte en un pilar igual de importante para culminar satisfactoriamente los proyectos.

En relación con las técnicas para desarrollar que utiliza Accenture que fueron expuestas y explicadas por el desarrollador con más experiencia, se dio a conocer que, desde este tipo de aplicaciones, es posible utilizar componentes WebView, los cuales ofrecen algunos beneficios, ya que, en lugar de realizar varias pantallas para dos sistemas operativos diferentes, las aplicaciones realizan un llamado a una aplicación web a través de una WebView, que resulta igual de funcional que una aplicación móvil. Anexo a ellos, según las condiciones, podría resultar más eficiente realizar un nuevo despliegue de una versión, sobre una aplicación web, a una actualización móvil que llegue a todos los usuarios finales, ya que intervienen las tiendas virtuales.

Cabe resaltar que, para la aplicación móvil y las páginas realizadas instanciadas a través de las WebView, se utilizaron bibliotecas de estilos que eran ampliamente utilizadas por desarrolladores a nivel global, lo que permitía optimizar el tiempo,

eliminando requerimientos relacionados a colores, tipografía, *layout* y demás. Sumado a ello, las interfaces de la aplicación las realizaban diseñadores expertos en el tema, a través de una aplicación web de diseño, donde se exponía claramente el flujo del usuario, mensajes de validación y los componentes.

Se resaltó la importancia de desarrollar la aplicación bancaria nativamente por la identificación de requisitos y la inclusión de algunos otros en un futuro, favoreciendo aspectos relacionados a la seguridad, escalabilidad, practicidad y demás. Algunos ejemplos son los siguientes: Para una aplicación es imperativo que no se puedan tomar capturas de pantalla dentro de ella, es necesario bloquear la aplicación en segundo plano para que no sea accesible por otras aplicaciones, gestionar hardware dentro de la aplicación. Estas funcionalidades se pueden encontrar en diversas bibliotecas para acceder a estos servicios más eficientemente.

Debido a una de las dificultades mencionadas anteriormente en relación con las pruebas unitarias y de integración, uno de los desafíos más importantes consistió en adquirir rápidamente los conocimientos necesarios para empezar a aplicar TDD y realizar tareas cada vez mas complejas e importantes para el proyecto. De esta forma, se considera importante para la estimación de proyectos que incluyan practicantes, tener en cuenta que la curva de aprendizaje es considerablemente amplia y dificulta el acoplamiento inicial, ya que no en todas las universidades se enseñan estos temas.

Un desafío presente cuando se trabajó con el banco Icol consistía en el proceso complejo para lograr una comunicación efectiva con los encargados del cliente, lo cual dificultaba la puesta en practica de las metodologías ágiles, sumado a que asumían que utilizar *Jira* y reunirse de forma ocasional con el equipo, representa aplicar esta metodología. Sin embargo, esto va más allá de crear un backlog que gestiona únicamente Accenture y utilizar *Jira* como herramienta de gestión. Aunque desde el rol ejercido realizar una propuesta de mejora o alguna solución resultaba muy difícil, un desafío podría consistir en tratar de mejorar la comunicación y lograr una mejor inclusión al cliente dentro del proyecto.

Gracias al proceso de caracterización fue posible obtener una visión general de las responsabilidades adquiridas para realizar un efectivo acoplamiento a cada uno de los roles, sumado a ello, se observaron algunas ventajas y desventajas que se pueden presentar al trabajar aplicando ciertas tecnologías o metodologías. Por ejemplo: Se observó que dividir un proyecto de gran magnitud en subproyectos, favorece en muchos aspectos a la entrega de un producto de calidad, aportando que cada subproyecto se haya analizado, planeado, ejecutado y probado de manera correcta en los tiempos estimados. La división también permite gestionar fácilmente cada fracción y aspectos relacionados, como el personal, dificultades no estimadas, errores reportados o fallas en la estimación, y hace posible que los tiempos se puedan ajustar y la ruta crítica del cronograma no se atrase.

En paralelo al trabajo realizado con el banco Cubo (Accenture Forms), que permitió agilizar ciertos procesos al grupo de gestión de proyectos, se participó y contribuyó, en conjunto con el grupo multidisciplinario a una serie de actividades, resolviendo necesidades relacionadas con la estimación de requisitos, monitoreo de cronogramas, levantamiento de actas, asistencia y organización de reuniones de seguimiento. Estas y otras actividades se coordinaron en conjunto con un practicante quien gestionaba la mayoría de esas actividades, y el Application Manager, quien debía responder ante el cliente.

En conjunto con la experiencia obtenida en los inicios de la práctica, y los conocimientos y habilidades adquiridas en la carrera universitaria, fue posible construir un producto (Accenture Forms) realizado desde su inicio hasta su final, donde se evidencia la experiencia obtenida, ya que se pusieron en práctica todas las recomendaciones para obtener una aplicación con una alta satisfacción del cliente, construido pensando en la calidad y escalabilidad del mismo.

Referencias bibliográficas

[1] «Accenture | Our purpose». https://www.accenture.com/us-en/about/company-index (accedido ago. 02, 2021).
[2] «Accenture | Colombia | Que Haya Cambio». https://www.accenture.com/co-es (accedido ago. 02, 2021).
[3] «Digital Transformation Initiative Aviation, Travel and Tourism», p. 40.
[4] O. Abbosh y K. Bissell, «Securing the Digital Economy: Reinventing the Internet for Trust», p. 49.
[5] «Awards and Recognition for Accenture». https://www.accenture.com/us-en/about/awards-recognition (accedido ago. 02, 2021).
[6] L. Williams, «Agile Software Development Methodologies and Practices», en *Advances in Computers*, vol. 80, M. V. Zelkowitz, Ed. Elsevier, 2010, pp. 1-44. doi: 10.1016/S0065-2458(10)80001-4.
[7] E. Guerra y M. Aniche, «Chapter 9 - Achieving quality on software design through test-driven development», en *Software Quality Assurance*, I. Mistrik, R. Soley, N. Ali, J. Grundy, y B. Tekinerdogan, Eds. Boston: Morgan Kaufmann, 2016, pp. 201-220. doi: 10.1016/B978-0-12-802301-3.00009-0.
[8] N. Lalband y D. Kavitha, «Software Development Technique for the Betterment of End User Satisfaction using Agile Methodology», *TEM J.*, vol. 9, pp. 992-1002, ago. 2020, doi: 10.18421/TEM93-22.
[9] F. Khan, S. Rasheed, M. Alsheshtawi, T. Ahmed, y S. Jan, «A Comparative Analysis of RAD and Agile Technique for Management of Computing Graduation Projects», *Cmc - Tech Sci. Press.*, vol. 64, pp. 777-796, jun. 2020, doi: 10.32604/cmc.2020.010959.
[10] L. Lebdeh, A. Qasim, y F. Kharbat, «Implementing Agility in Large Software Development Projects», *TEM J.*, pp. 1285-1294, ago. 2020, doi: 10.18421/TEM93-58.
[11] A. Santos, S. Vegas, F. Uyaguari, O. Dieste, B. Turhan, y N. Juristo, «Increasing validity through replication: an illustrative TDD case», *Softw. Qual. J.*, vol. 28, n.º 2, pp. 371-395, jun. 2020, doi: 10.1007/s11219-020-09512-3.
[12] N. Borle, M. Feghhi, E. Stroulia, R. Greiner, y A. Hindle, «Analyzing the effects of test driven development in GitHub», may 2018, pp. 1062-1062. doi: 10.1145/3180155.3182535.
[13] A. Roman y M. Mnich, «Test-driven development with mutation testing – an experimental study», *Softw. Qual. J.*, vol. 29, n.º 1, pp. 1-38, mar. 2021, doi: 10.1007/s11219-020-09534-x.
[14] V. Bhadauria, R. Mahapatra, y S. Nerur, «Performance Outcomes of Test-Driven Development: An Experimental Investigation», *J. Assoc. Inf. Syst.*, vol. 21, pp. 1045-1071, jul. 2020, doi: 10.17705/1jais.00628.
[15] A. Dookhun y L. Nagowah, «Assessing The Effectiveness Of Test-Driven Development and Behavior-Driven Development in an Industry Setting», dic. 2019, pp. 365-370. doi: 10.1109/ICCIKE47802.2019.9004328.
[16] J. L. del Val Román, «Industria 4.0: la transformación digital de la industria», presentado en CONFERENCIA DE DIRECTORES Y DECANOS DE INGENIERÍA INFORMÁTICA, 2016.
[17] H. Kir y N. Erdogan, «A knowledge-intensive adaptive business process management framework», *Inf. Syst.*, vol. 95, p. 101639, ene. 2021, doi: 10.1016/j.is.2020.101639.
[18] P. Calvo, «Etificación, la transformación digital de lo moral», *Kriter. Rev. Filos.*, vol. 60, pp. 671-688, dic. 2019, doi: 10.1590/0100-512x2019n14409pc.
[19] D. Rogers, *The Digital Transformation Playbook*. Columbia University Press, 2016. Accedido: ago. 07, 2021. [En línea]. Disponible en: https://www.degruyter.com/document/doi/10.7312/roge17544/html
[20] I. Drigă y C. Isac, «E-banking services – features, challenges and benefits», *Ann. Univ. Petrosani Econ.*, vol. 14, n.º 1, pp. 49-58, 2014.
[21] J. C. Narváez Gutiérrez, V. Núñez Zapata, y D. V. Muñoz Marín, «Análisis del modelo de

banca móvil y confianza financiera en Colombia», Thesis, Universidad Santiago de Cali, 2020. Accedido: ago. 06, 2021. [En línea]. Disponible en: https://repository.usc.edu.co/handle/20.500.12421/4448
[22] M. D. Ionno y M. Michael, «Seguimiento de la Economía de las Aplicaciones en Colombia», *Progress. Policy Institude*, p. 12.
[23] R. Poblete y C. Andrés, «Evaluación de la seguridad de aplicaciones móviles bancarias», 2016, Accedido: ago. 07, 2021. [En línea]. Disponible en: http://repositorio.uchile.cl/handle/2250/144529
[24] T. Luo, H. Hao, W. Du, Y. Wang, y H. Yin, «Attacks on WebView in the Android system», dic. 2011, pp. 343-352. doi: 10.1145/2076732.2076781.
[25] J. Offutt y Y. Wu, «Modeling presentation layers of web applications for testing», *Softw. Syst. Model.*, vol. 9, n.º 2, pp. 257-280, abr. 2010, doi: 10.1007/s10270-009-0125-4.
[26] R. S. Cox, J. G. Hansen, S. D. Gribble, y H. M. Levy, «A safety-oriented platform for Web applications», en *2006 IEEE Symposium on Security and Privacy (S P'06)*, may 2006, p. 15 pp. - 364. doi: 10.1109/SP.2006.4.
[27] «Primeros pasos en React - Aprende sobre desarrollo web | MDN». https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started (accedido ago. 07, 2021).
[28] D. T. Kudiabor, «State management with React-Redux», fi=AMK-opinnäytetyö|sv=YH-examensarbete|en=Bachelor's thesis|, Centria University of Applied Sciences, 2020. Accedido: ago. 07, 2021. [En línea]. Disponible en: http://www.theseus.fi/handle/10024/355184
[29] «¿Qué es Software as a Service (SaaS)?», *¿Qué es SaaS?* https://www.oracle.com/co/applications/what-is-saas/ (accedido ago. 07, 2021).
[30] «Uso y límites», *Firebase*. https://firebase.google.com/docs/firestore/quotas?hl=es (accedido jul. 16, 2021).
[31] «Usage Limits | Sheets API», *Google Developers*. https://developers.google.com/sheets/api/reference/limits?hl=es (accedido jul. 16, 2021).
[32] «Presentando JSX – React». https://es.reactjs.org/docs/introducing-jsx.html (accedido jul. 18, 2021).
[33] «CDN - MDN Web Docs Glossary: Definitions of Web-related terms | MDN». https://developer.mozilla.org/en-US/docs/Glossary/CDN (accedido jul. 20, 2021).